

Korisničke funkcije (Use Case)

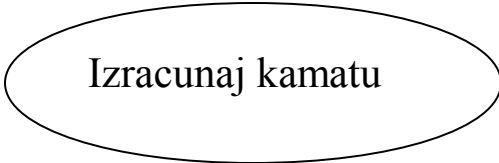
Mogli bismo ih nazvati i korisnički slučajevi ili slučaj upotrebe.

Korisničke funkcije služe da se prikaže željeno ponašanje sistema koji se razvija. Ne mora da se navede kako se funkcije realizuju. Služe za sporazumevanje projektanata i krajnjih korisnika sistema.

Korisničke funkcije se mogu primeniti na ceo sistem ili na njegove delove.

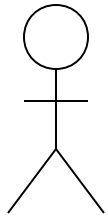
Korisnička funkcija predstavlja opis niza akcija, uključujući i varijacije, koji sistem izvodi.

Korisnička funkcija se grafički predstavlja elipsom. Ona uvek ima ime koje se upisuje u elipsu.



Izracunaj kamatu

Korisničke funkcije se kombinuju sa **akterima (izvođačima)**. Izvođač je najčešće čovek, ali može biti hardverski uređaj ili čak drugi sistem. Akter se predstavlja često grafičkim simbolom:



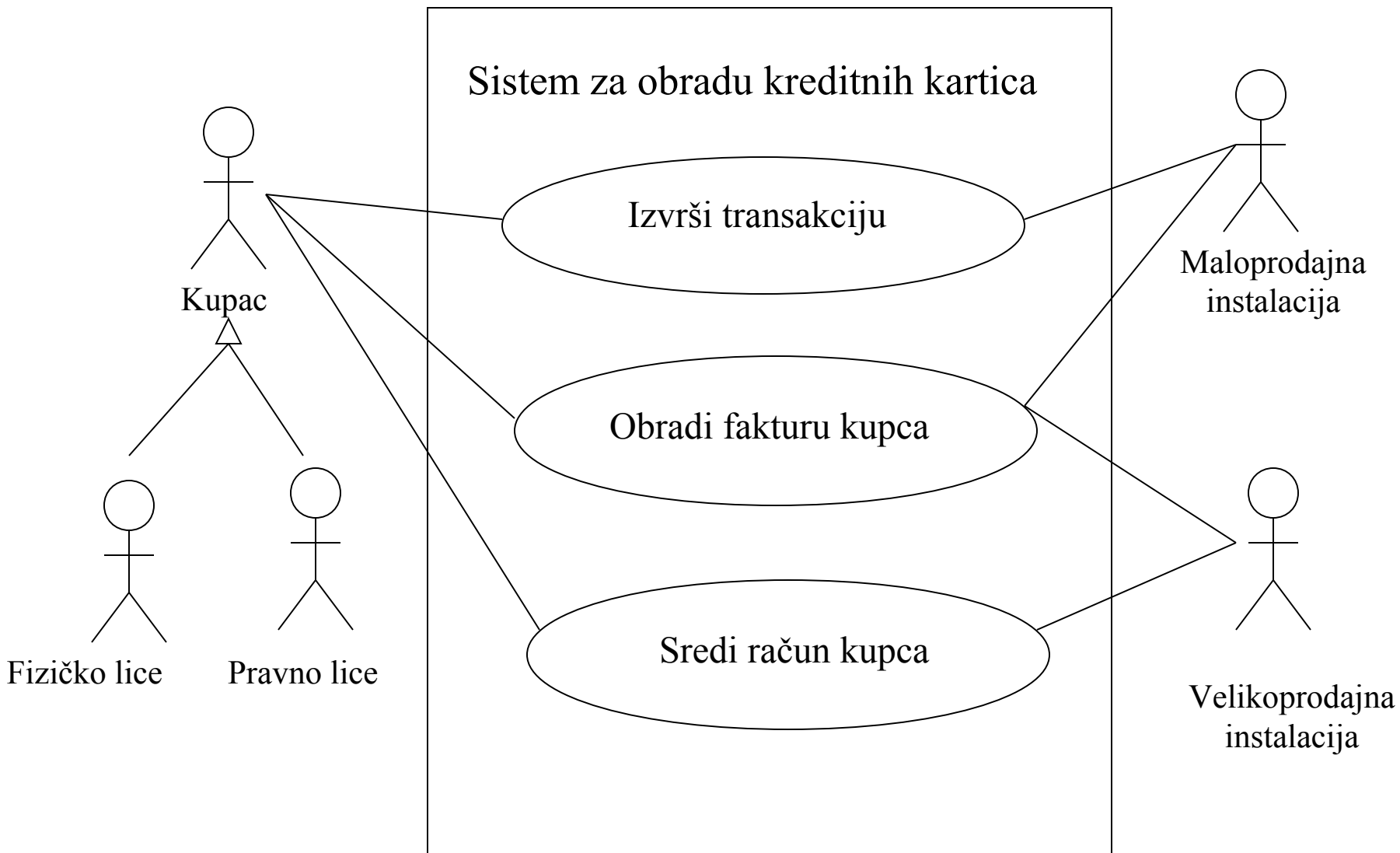
U uskoj vezi sa korisničkim funkcijama je pojam **scenario**. Scenario predstavlja niz koraka kojima se opisuje interakcija između korisnika i sistema.

Primer: Obrada kreditne kartice kupca

Korisnička funkcija je skup scenarija povezanih tako da se ostvari korisnički cilj (nekog aktera). Sadržaj korisničke funkcije se često opisuje tekстом preko niza koraka.

Dijagrami korisničkih funkcija sastoje se iz: korisničkih funkcija, aktera i relacija. Mogu sadržati komentare i pakete.

Opisuju spolja vidljive servise koje sistem omogućava.



Dijagrami stanja (mašine)

Dijagrami stanja služe za modelovanje dinamičkih aspekta sistema. Dijagrami stanja prikazuju se konačnim automatima. Dijagram aktivnosti je specijalan slučaj dijagrama stanja u kojem je prisutna većina stanja aktivnosti.

Stanje je situacija u životu objekta kada on zadovoljava neki uslov. Objekat ostaje u nekom stanju u končnom vremenskom intervalu.

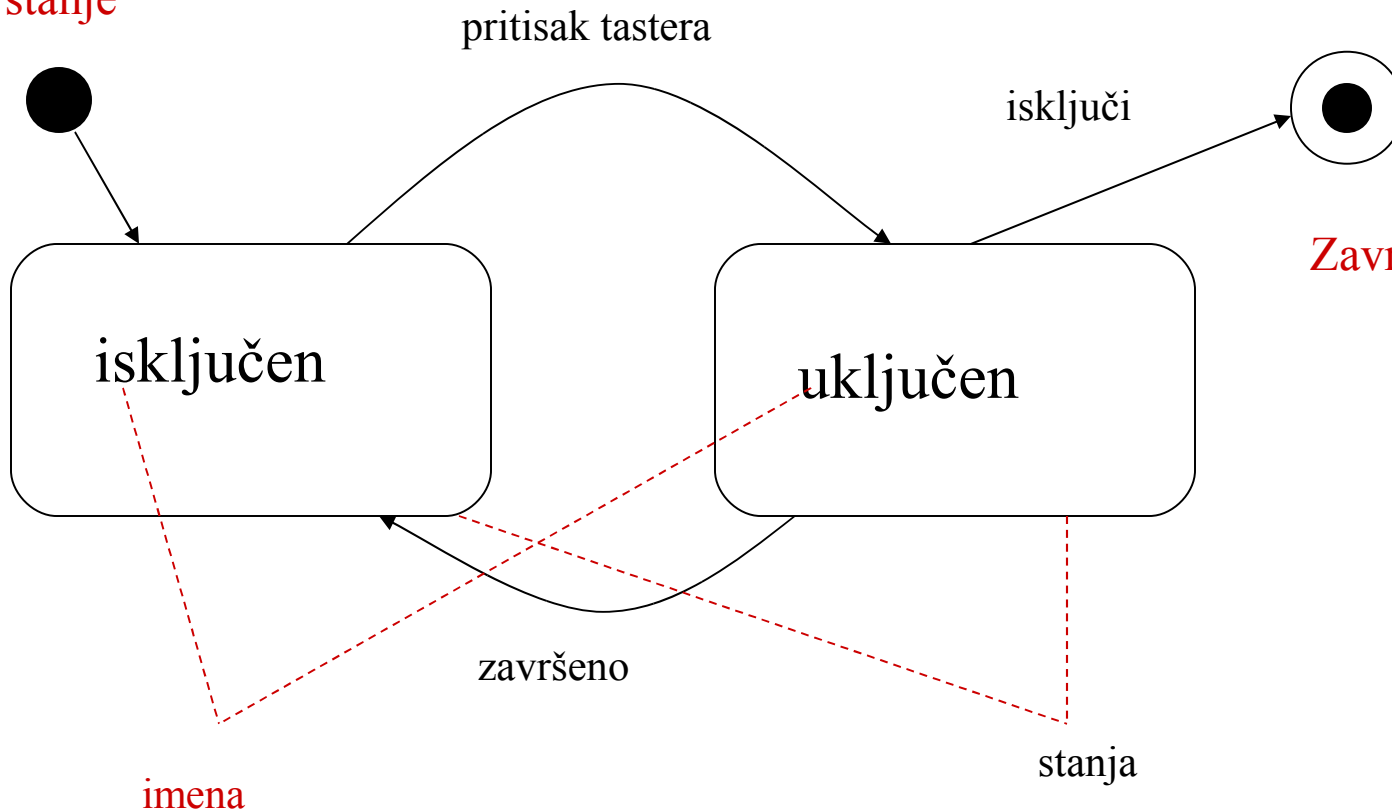
Dijagram stanja pokazuje odvijanje upravljanja od stanja do stanja. Obično modeluju objekte koji reaguju.

Tranzicija pokazuje kretanje od jednog stanja ka drugom. To je relacija između 2 stanja koja kaže da će objekat u jednom stanju izvršiti neke akcije pa će ući u drugo stanje.

Tranzicija ima tri opciona dela: **pobuđen događaj**, **gard (zaštitni uslov)** i **aktivnost**.

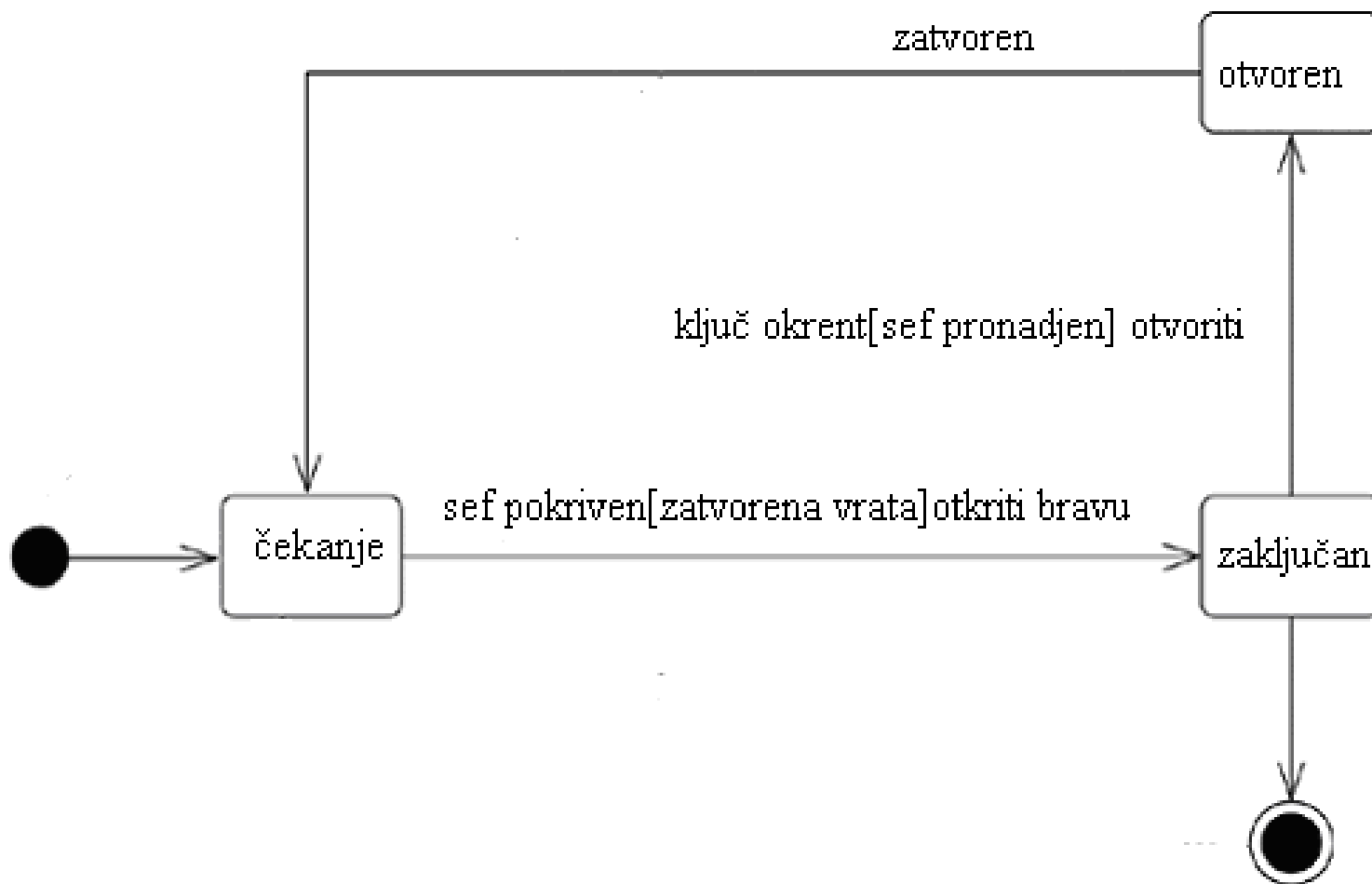
Na sledećoj slici prikazan je jednostavan dijagram stanja (nekog dugmeta).

Početno stanje

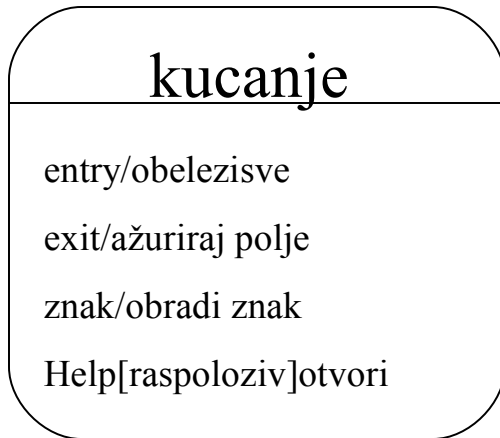


Završno stanje

Dijagram za opis stanja sefa.



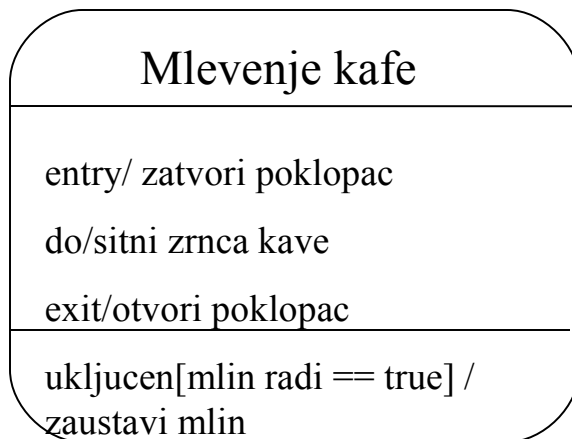
Stanje može reagovati na **unutrašnje aktivnosti**. Ovde se **entry**



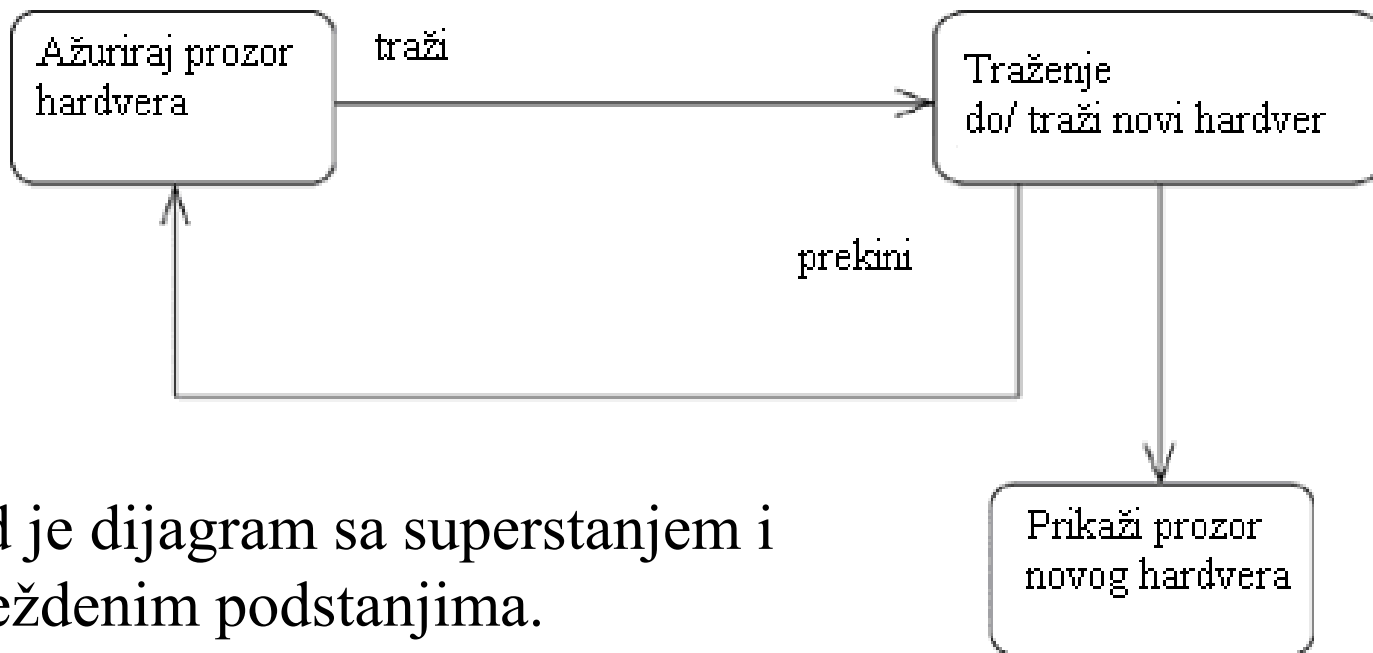
realizuje kad uđemo u stanje, a **exit** kad izlazimo iz stanja. Unutrašnje aktivnosti ne mogu da izazovu ulaznu i izlaznu aktivnost.

U dosadašnjim opisima objekti su mirovali i čekali neke događaje pre no što učine nešto.

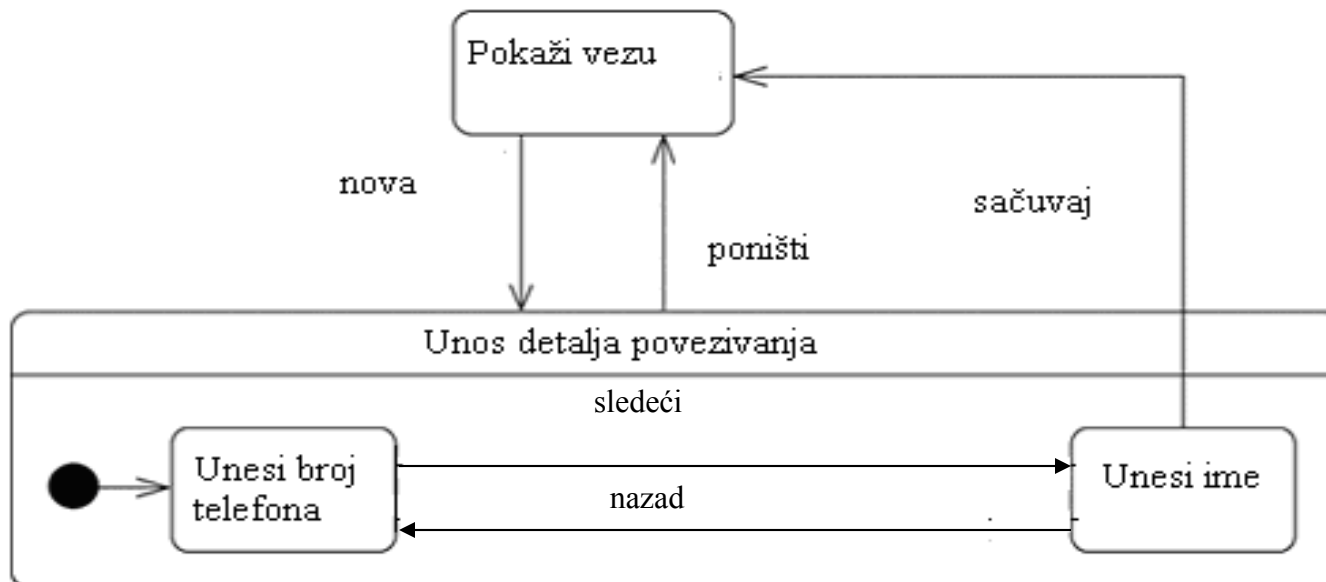
Može se opisati i stanje u kojem objekat nešto radi. To su stanja aktivnosti. Aktivnost se izražava u stanju rezervisanom rečiju **do /**.



Na sledećoj slici prikazan je dijagram stanja u kojem je jedno stanje aktivnosti.



Ispod je dijagram sa superstanjem i ugnježenim podstanjima.



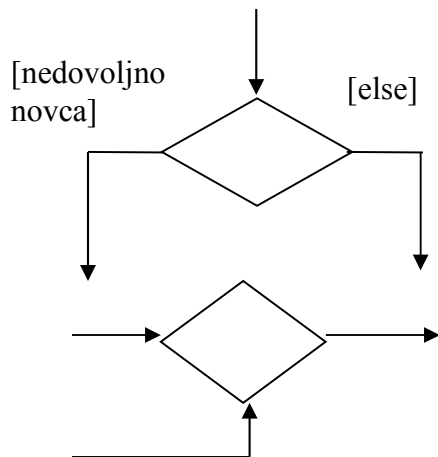
Dijagrami aktivnosti

Dijagrami aktivnosti služe za modelovanje dinamičkih aspekata sistema. Oni služe da prikažu: **proceduralnu logiku, poslovni proces ili tok posla**. Slični su blok-dijagramima za opis algoritama (dodatno, podržavaju **paralelno ponašanje**).

Dijagrami aktivnosti mogu se tretirati kao specijalan slučaj dijagrama stanja. U čvorovima ovog dijagrama prikazane su **akcije**. Akcije se implementiraju kao metodi klasa ili neke podaktivnosti.

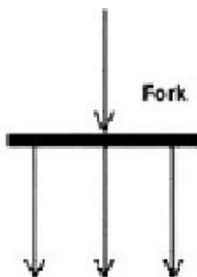
Dijagrami aktivnosti opisuju šta se radi, ali ne kažu ko šta radi. Ako želimo istaći ko šta radi (koja klasa je odgovorna za određenu akciju), podelićemo dijagram na particije.

Pore već ranije korišćenih simbola, ovde se koristi i niz novih.
Neki od njih su sledeći:

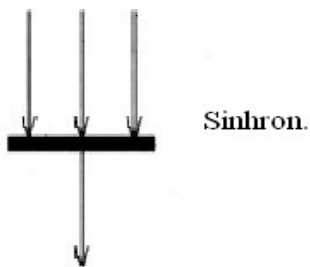
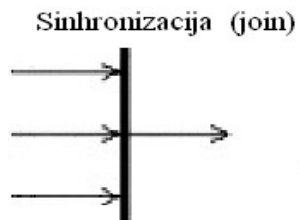


Grananje

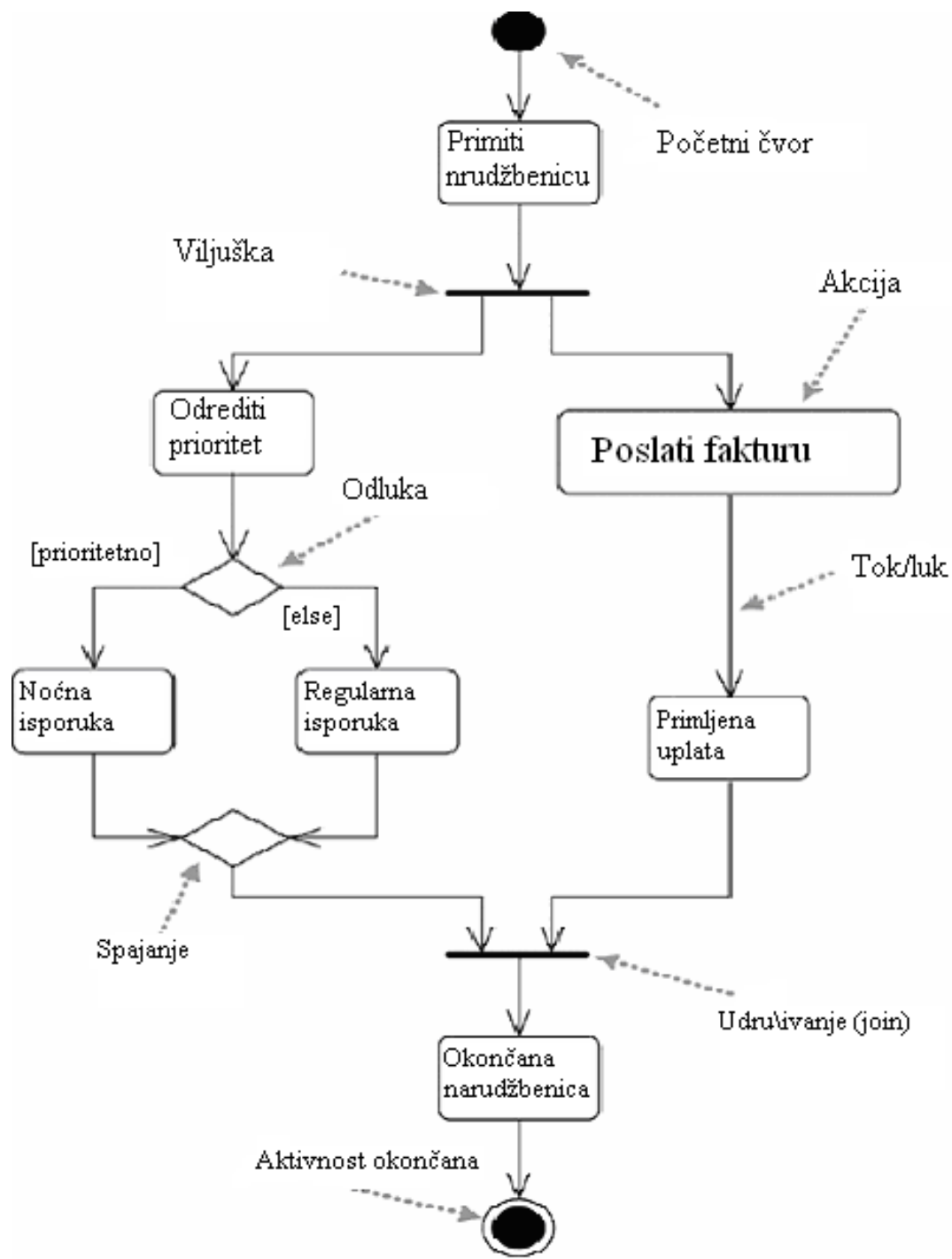
Spajanje



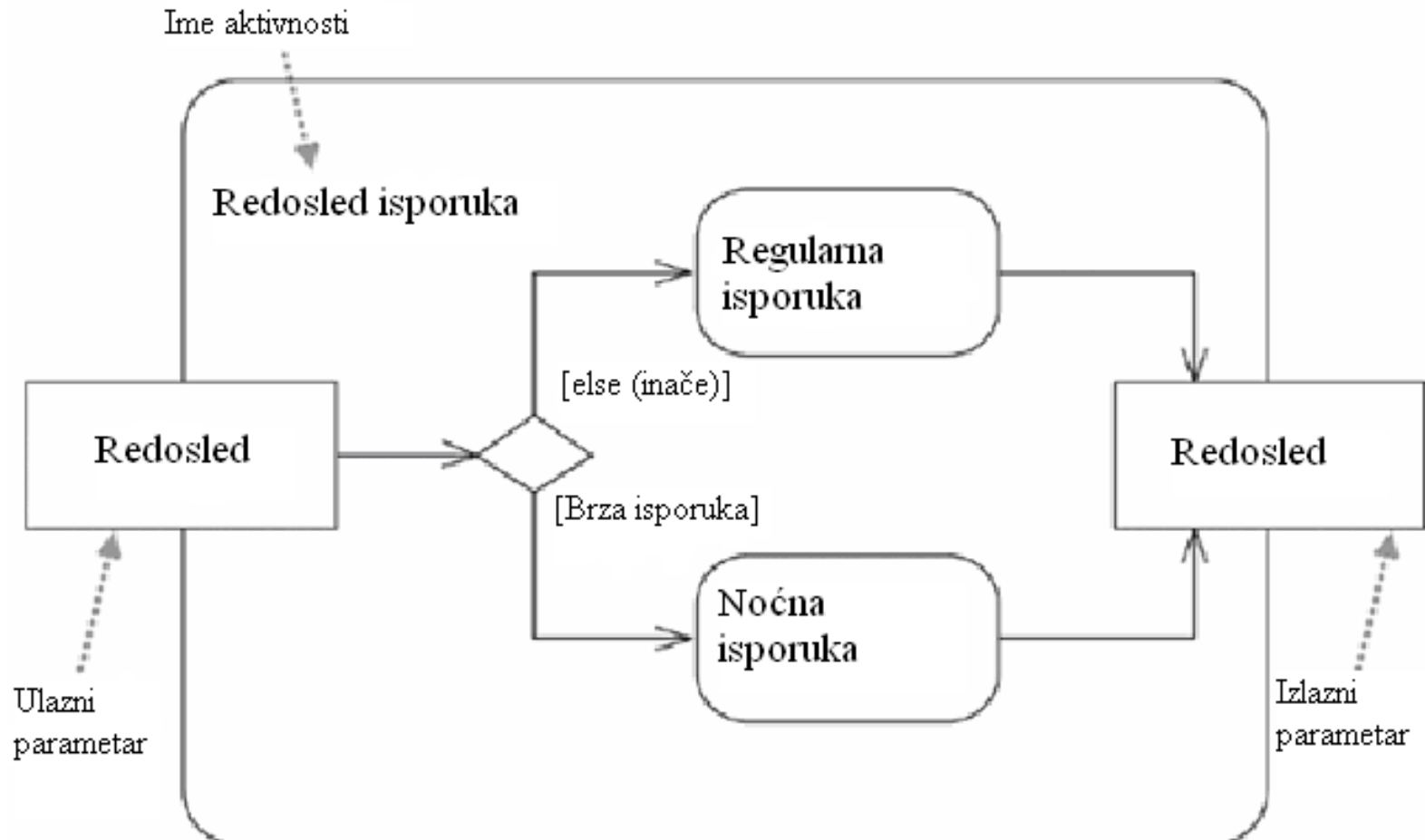
Inicijalizacija nekoliko
tredova ili paralelnih
procesa



Spajanje kontrola
korišćenjem sinhronizacije

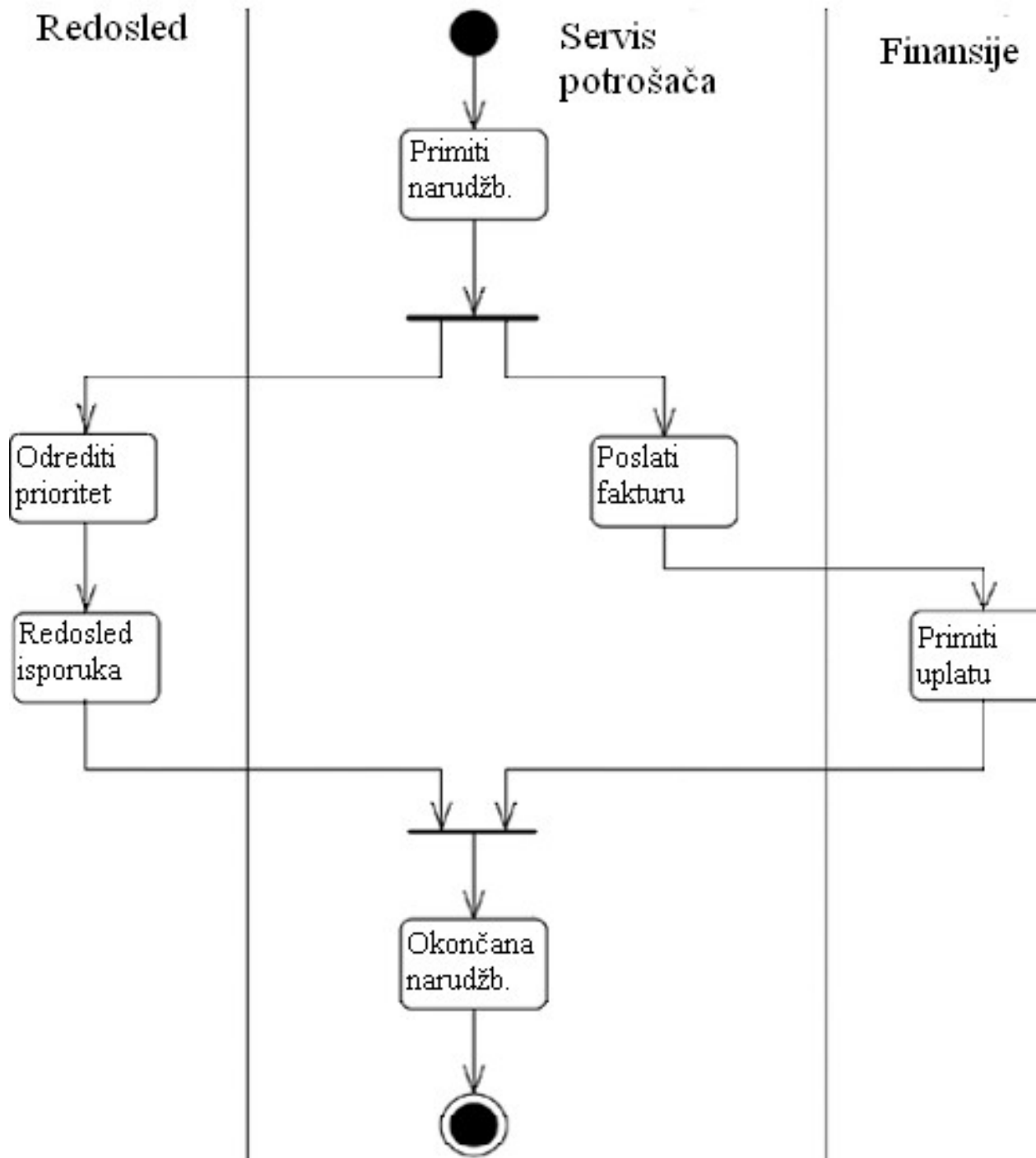


Akcije mogu biti razbijene na podakcije (odnosno, objedinjenje u jedan opšti čvor) kao u sledećem dijagramu.

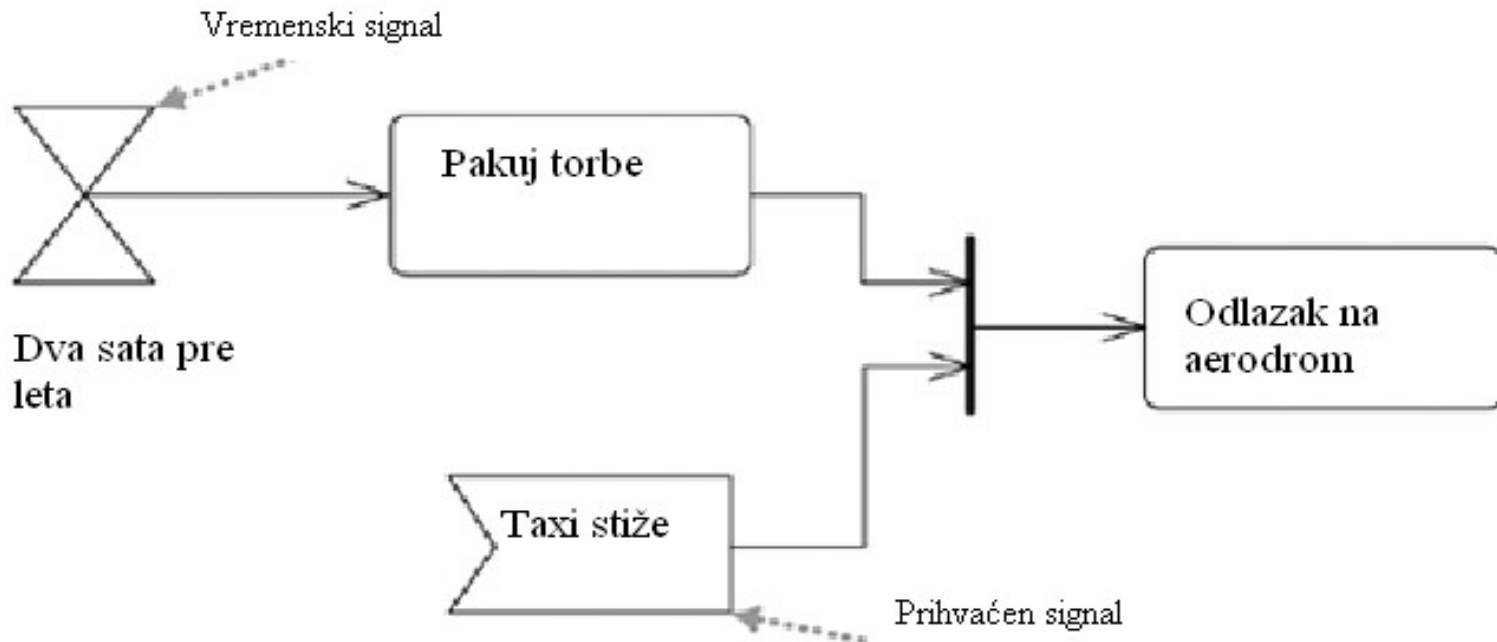


Sada se dijagram sa prethodnog slajda može preglednije predaviti (bez navođenja nekih detalja).

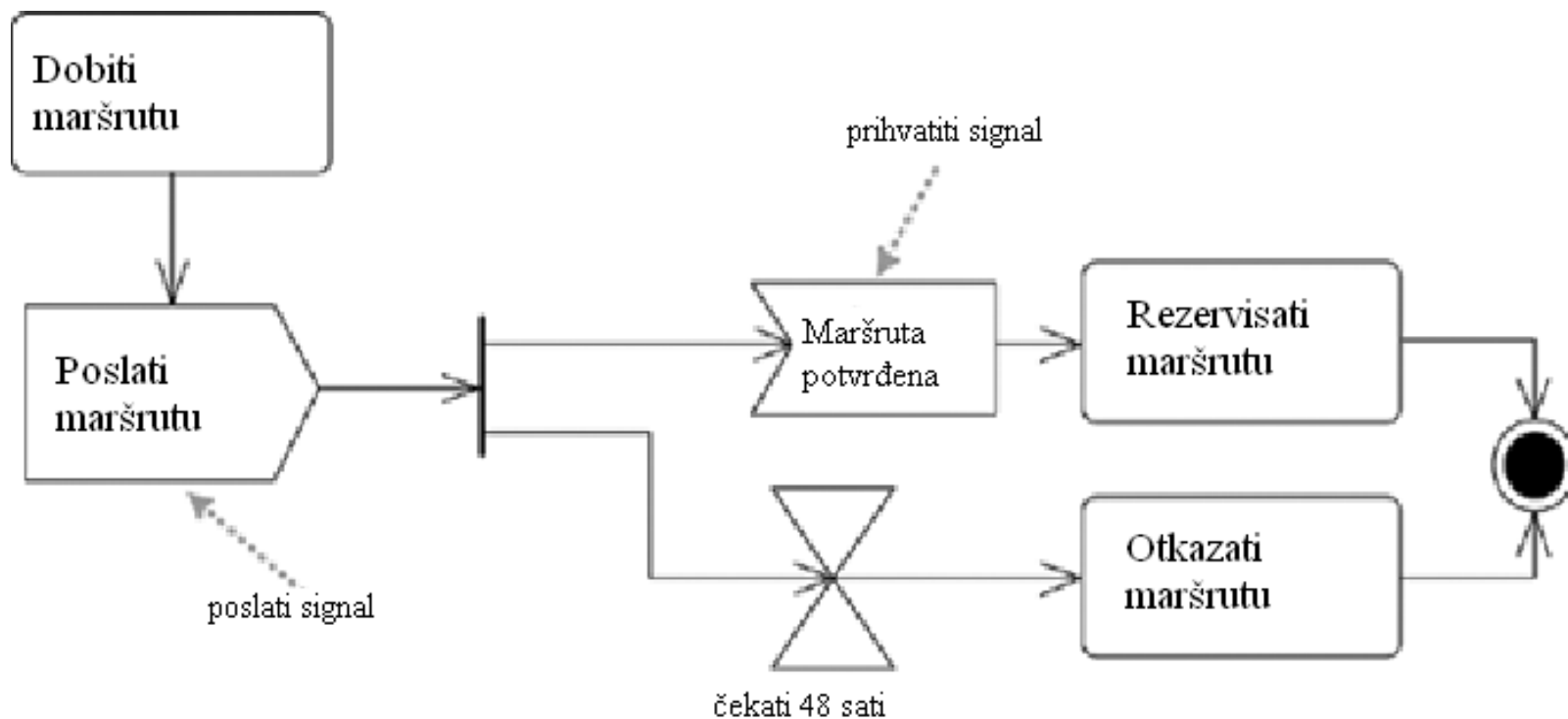
Deljenje dijagrama aktivnosti na particije. Iznad su ispisane zdužene klase za pojedine akcije. Umesto detaljnog prikaza, iskošćen je opšti čvor sa prethodne strane.



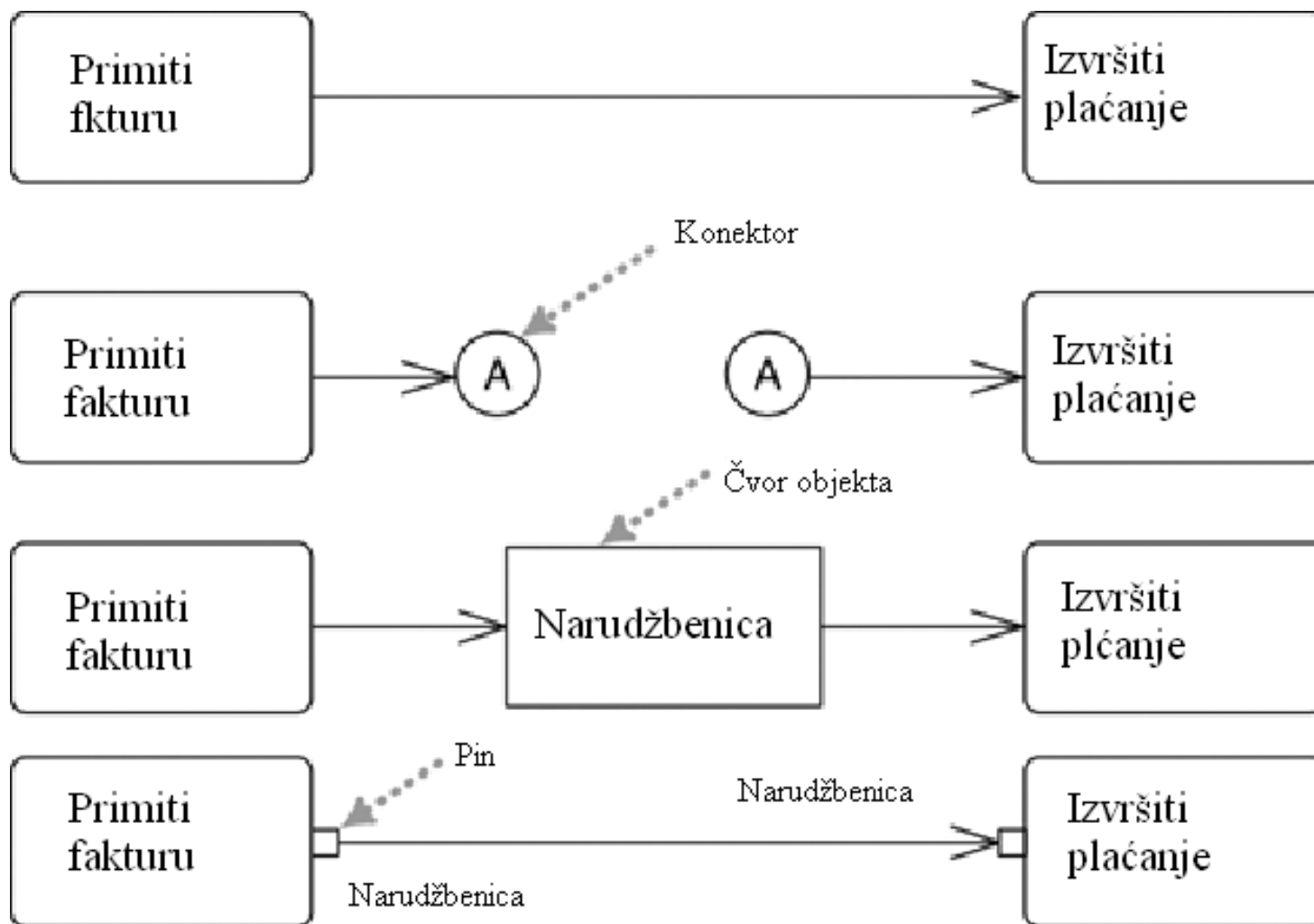
Pomoću dijagrama aktivnosti mogu se opisivati i događaji izazvani **slanjem nekih signala**. Signal pokazuje da aktivnost prima događaj iz nekog spoljašnjeg procesa. Aktivnost stalno osluškuje da li se pojavio signal, a dijagram opisuje reakciju kada se pojavi signal. Na primer, vremenski signal se pojavljuje kada protekne neko vreme (sekund, dan, nedelja ,...)



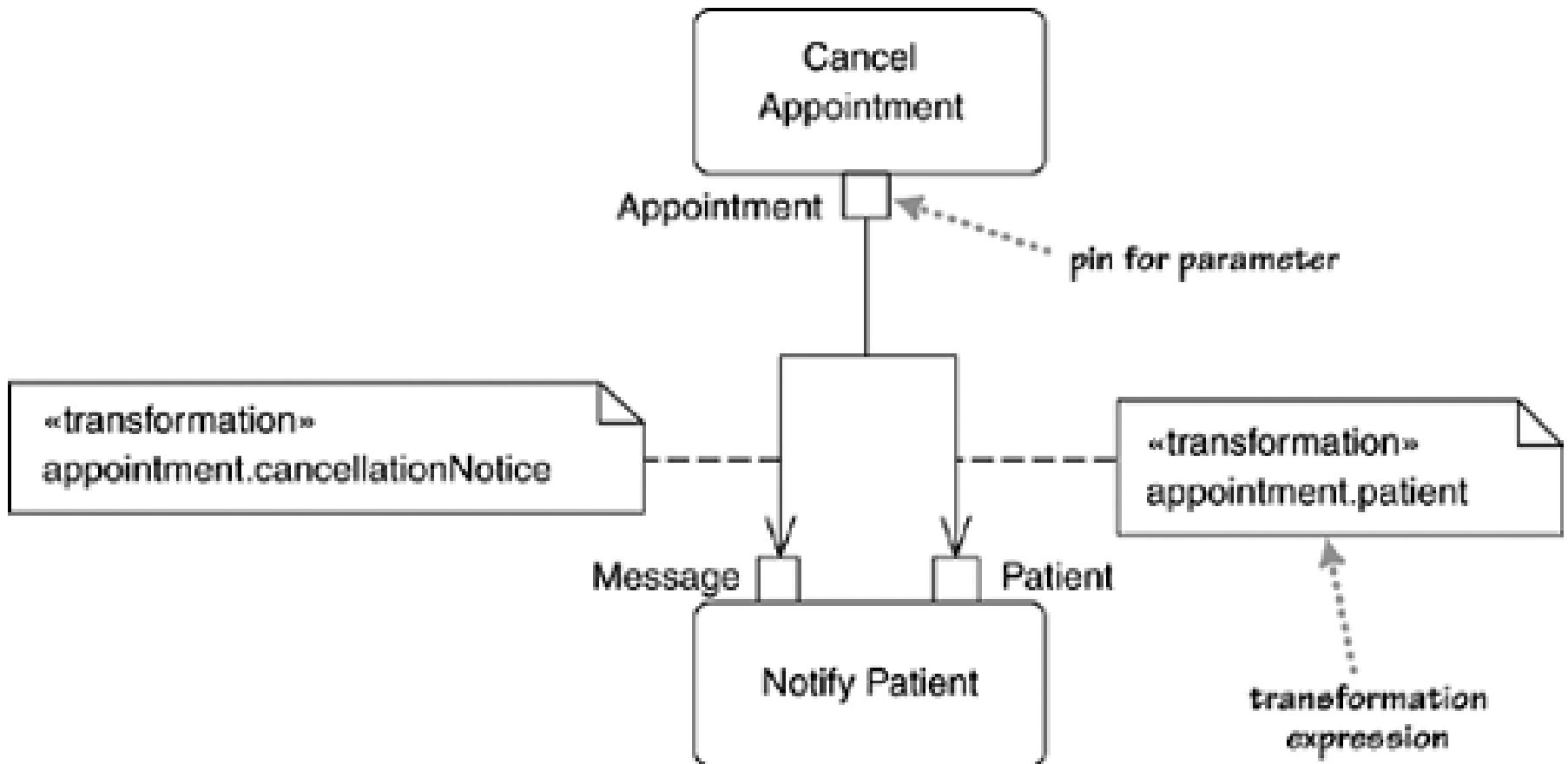
Kao što se može opisivati primanje signala, može se opisivati i njihovo **slanje**. Ovo je korisno kada šaljemo poruku i čekamo odgovor pre nego nastavimo.



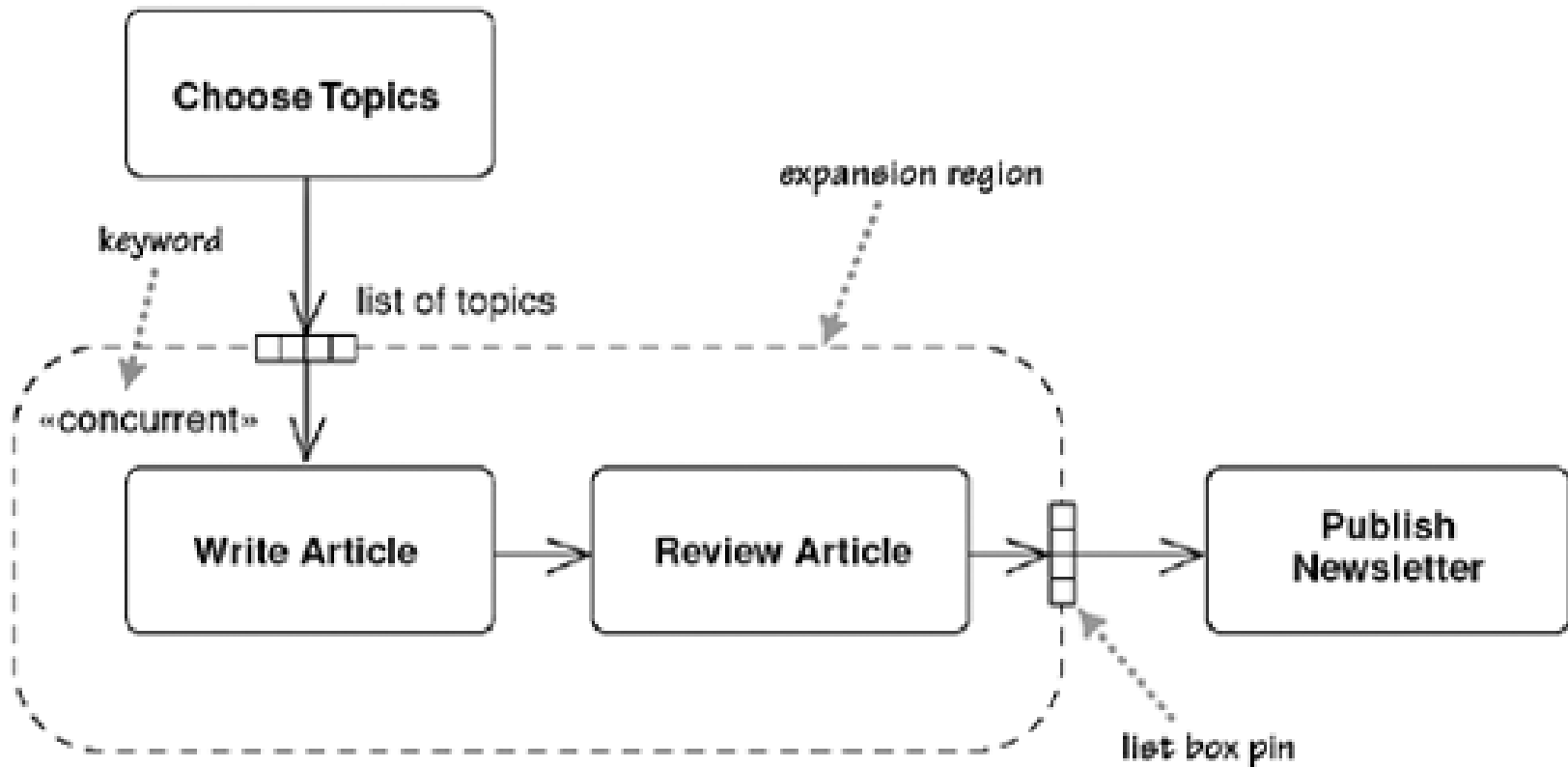
U UML-u 2 se koriste reči **TOK** i **LUK** da se opiše veza između 2 akcije. Lukovi (tokovi) se mogu prikazivati na razne načine. Na sledećoj slici su prikazana 4 načina.



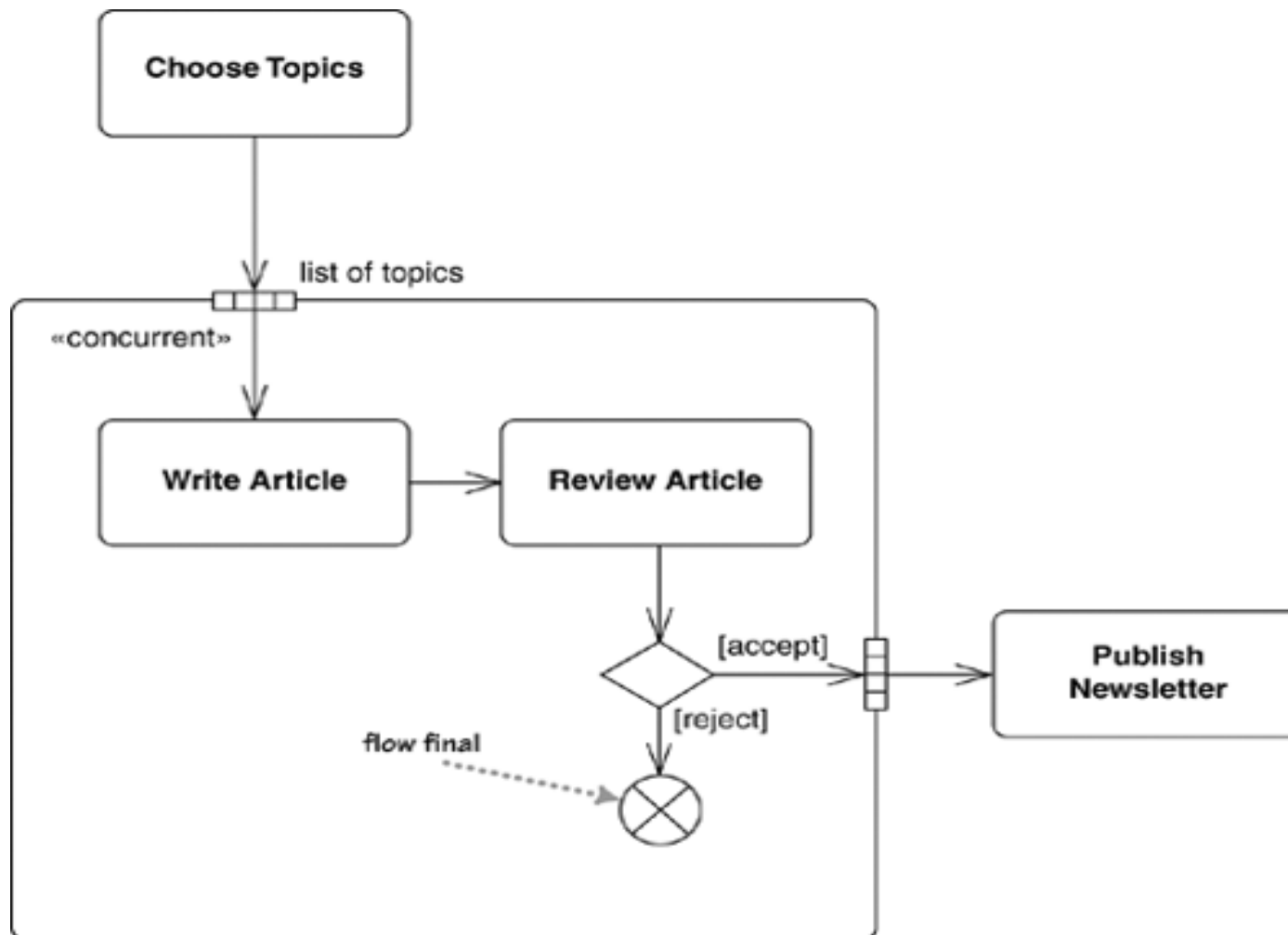
Akcije mogu imati parametre, kao i metodi. Parametri ne moraju da se prikazuju preko dijagrama aktivnosti, ali mogu. To se čini preko **pinova**. Akcija se razbija tako da pinovi odgovaraju parametar-boksovima na rastavljenom dijagramu.



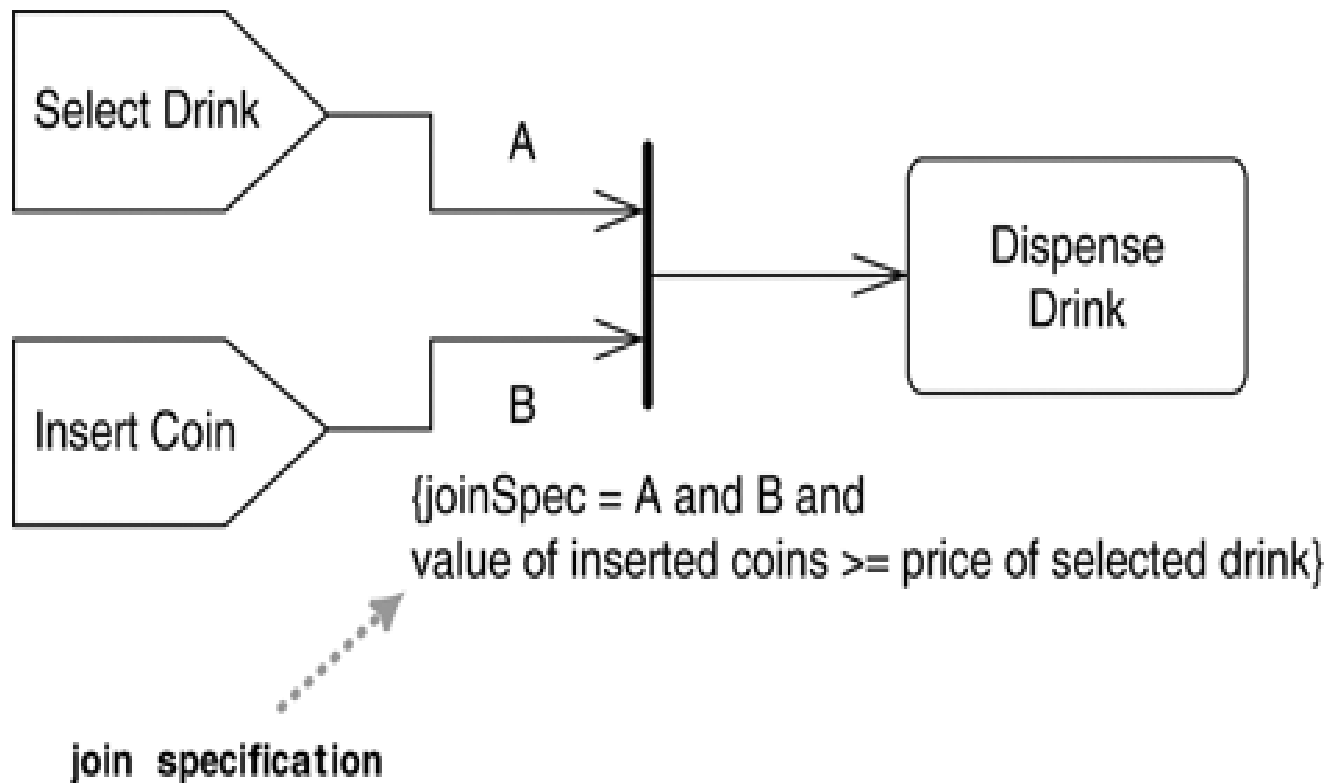
Ako jedan akcioni izlaz utiče na višestruko pozivanje druge akcije, najbolji način da se to izrazi je preko **ekspanzionih regiona**. Ekspanzioni region označava zonu dijagrama aktivnosti gde se akcije dešavaju jedanput za više stavki kolekcije.



Kada postoji više tokova (kao kod ekspanzionih regiona), neki tokovi se mogu okončati, a da se ne okonča cela aktivnost. To se opisuje na poseban način tzv. **finalnim stanjem**.

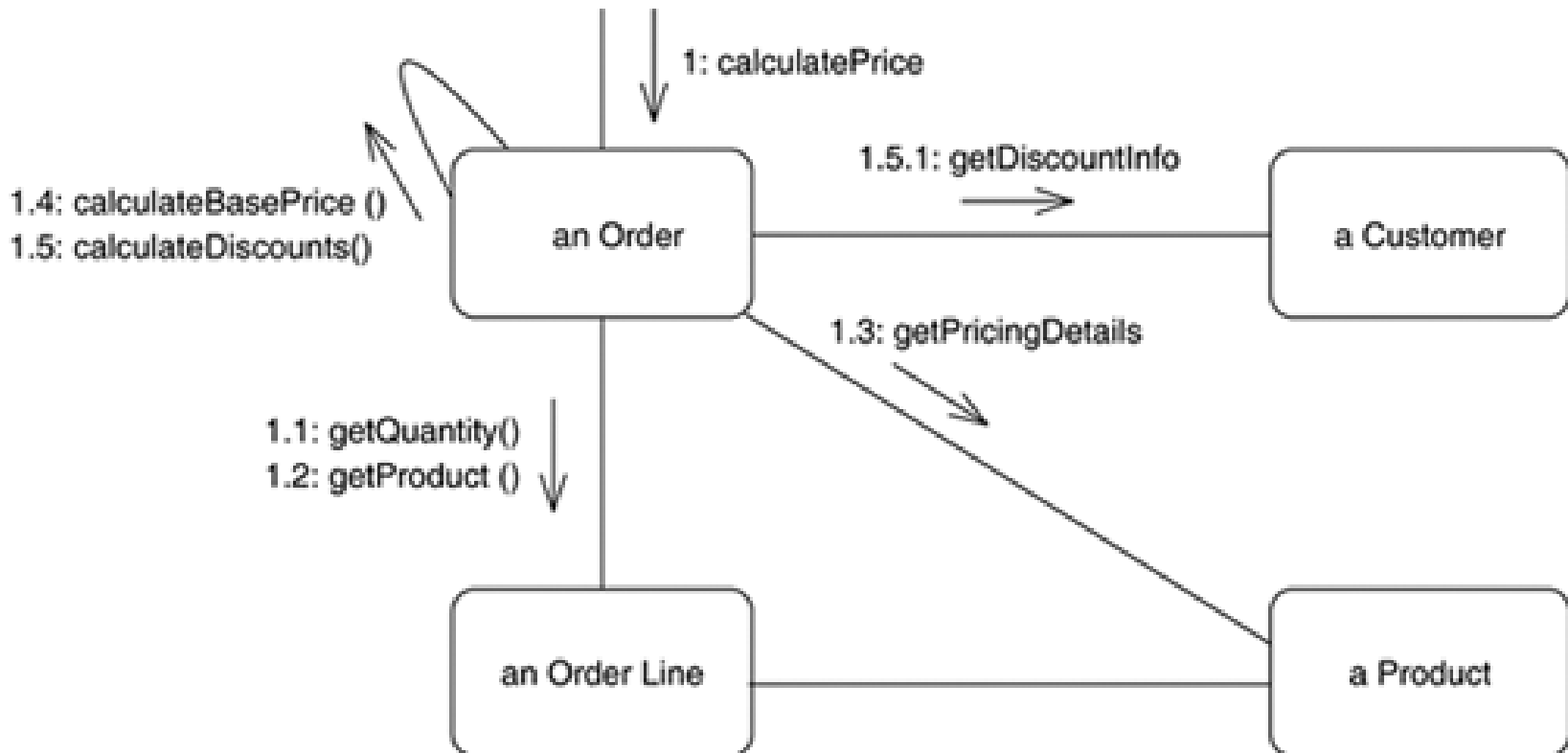


Specifikacija udruživanja (join specification) je Bulov izraz koji se pridružuje simbolu za združivanje (join) paralelnih procesa.



Komunikacioni dijagrami

Komunikacioni dijagrami predstavljaju jednu vrstu interakcionih dijagrama sa naglaskom na povezanosti podataka između raznih učesnika u interakciji. Primer jednog komunikacionog dijagrama je na slici. Decimalno označavanje metoda je novina u UML 2.

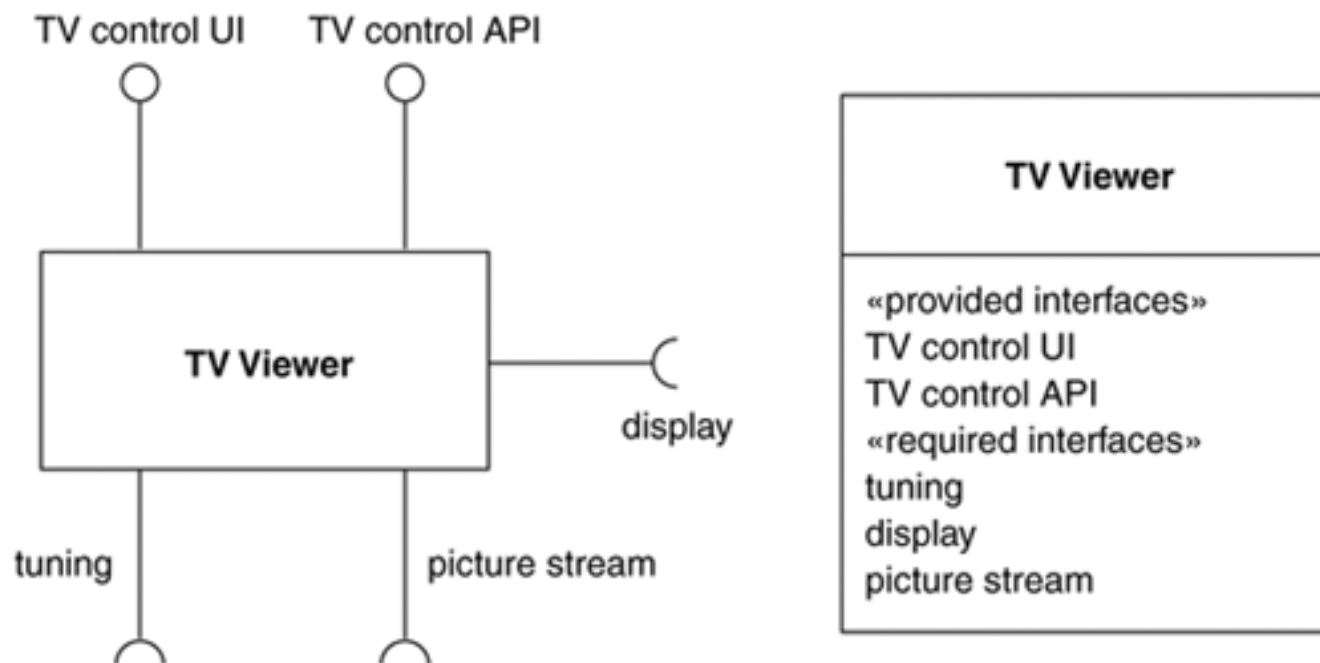


Preko komunikacionih dijagrama ne može se precizno izraziti kontrolna logika. Na primer, nema posebne notacije za kreiranje i brisanje objekata, ali se ključne reči *kreiraj* i *obriši* upotrebljavaju u ovim dijagramima.

Pored cifara, u porukama s emogu koristiti i slova. Preko njih se pogodno opisuju tredovi.

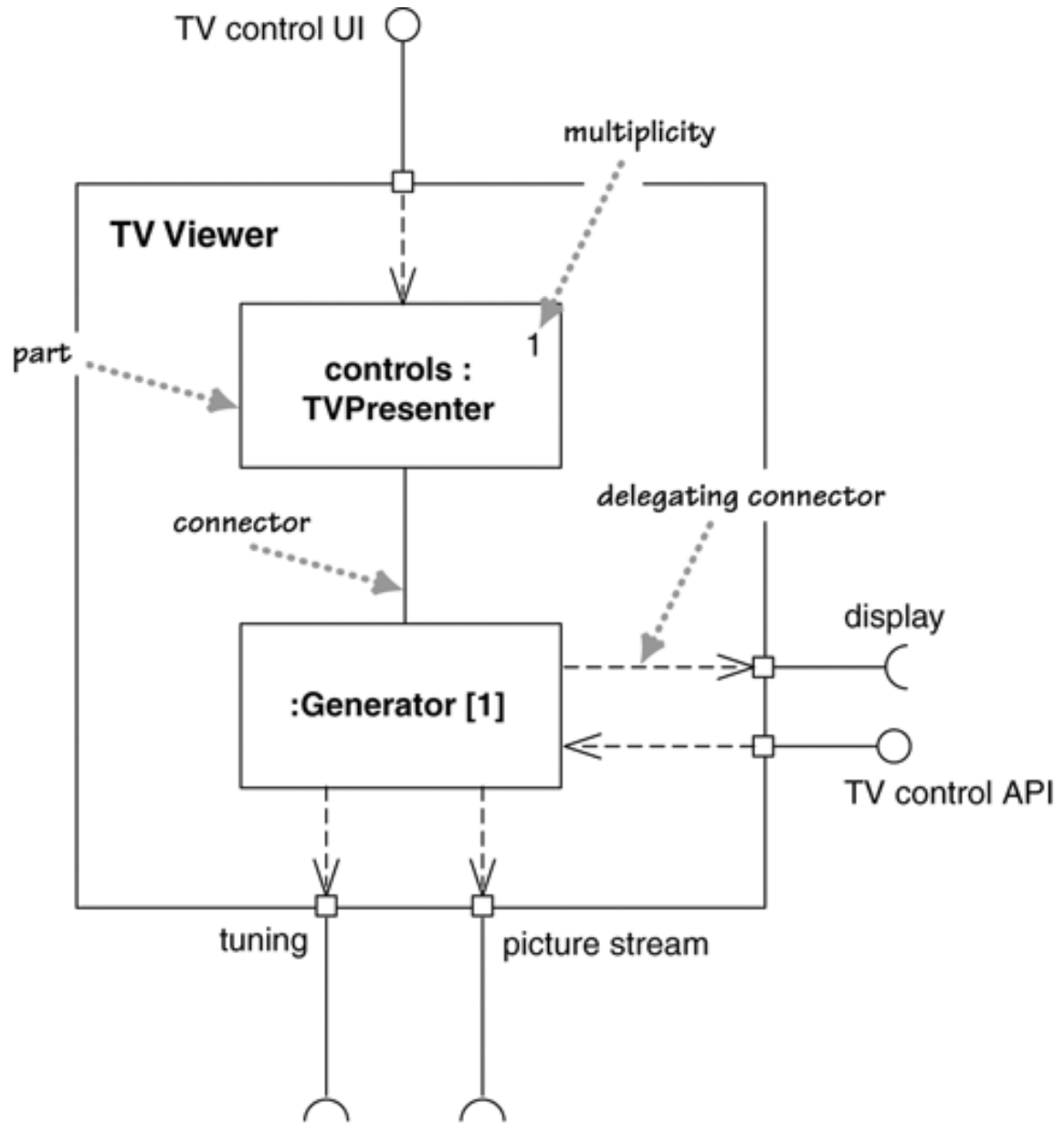
Kompozitne strukture

Ovo je značajna novina u UML 2. Dozvoljava hijerarhijsko rasčlanjivanje klase na unutrašnje strukture. Dakle, jedan kompleksan objekat može se razbiti u jednostavnije delove.



Ova klasa se može razložiti na 2 potklase, kao na sledećoj slici.

Za razliku od paketa koju pokazuju grupisanje za vreme kompajliranja, kompozitne strukture pokazuju grupisanje za vreme izvršavanja. Mogu se koristiti u komponentnim dijagramima.



Komponentni dijagrami

Komponente su delovi sistema. Teško je kazati šta je razlika između komponente i regularne klase. U UML-u se pravi razlika.



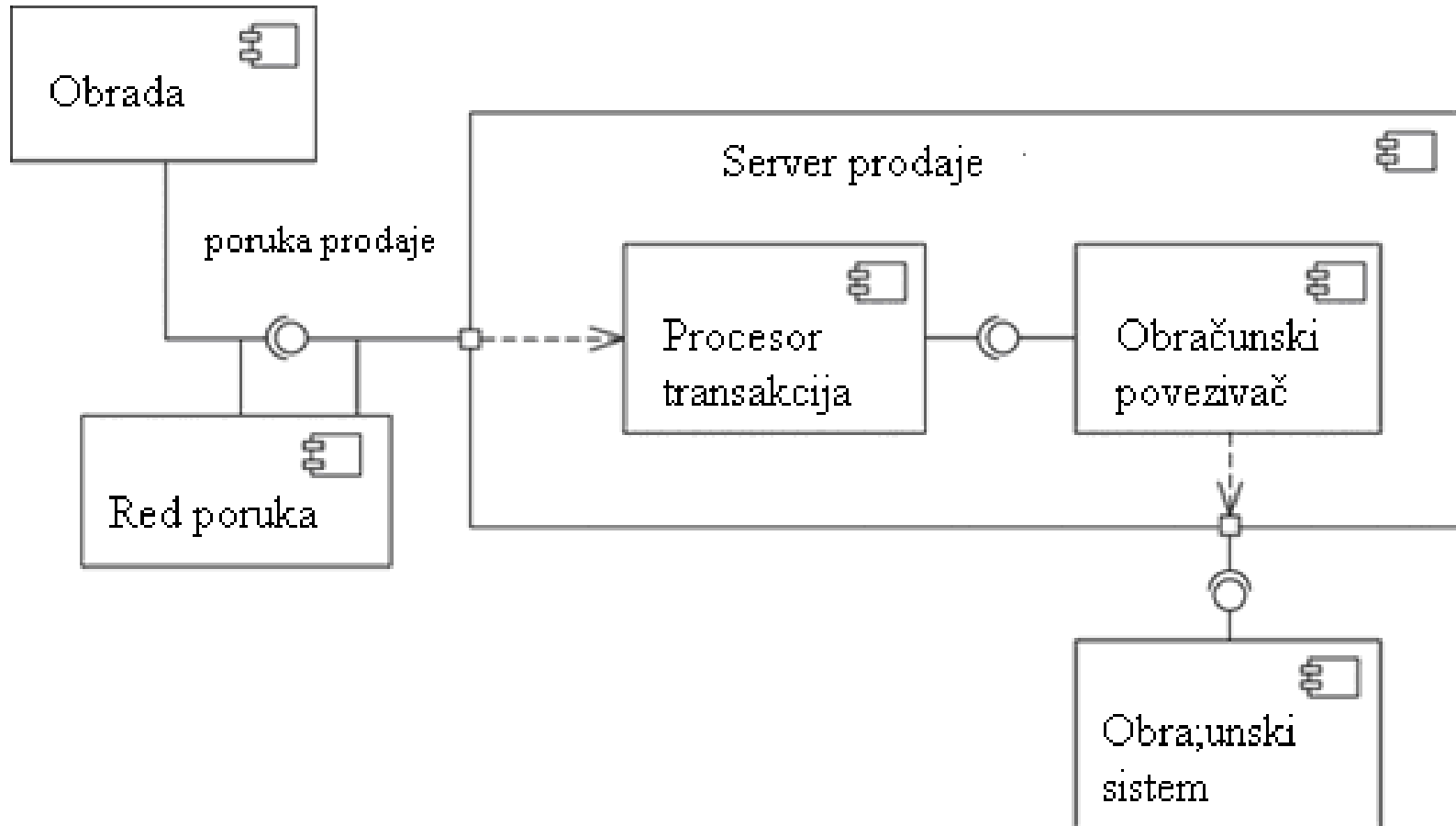
UML 1 notacija



UML 2 notacija

Komponente se povezuju preko implementiranih i zahtevanih interfejsa (kao i klase).

Primer jednog komponentnog dijagrama prikazan je na sledećoj slici.



Kolaboracioni dijagrami

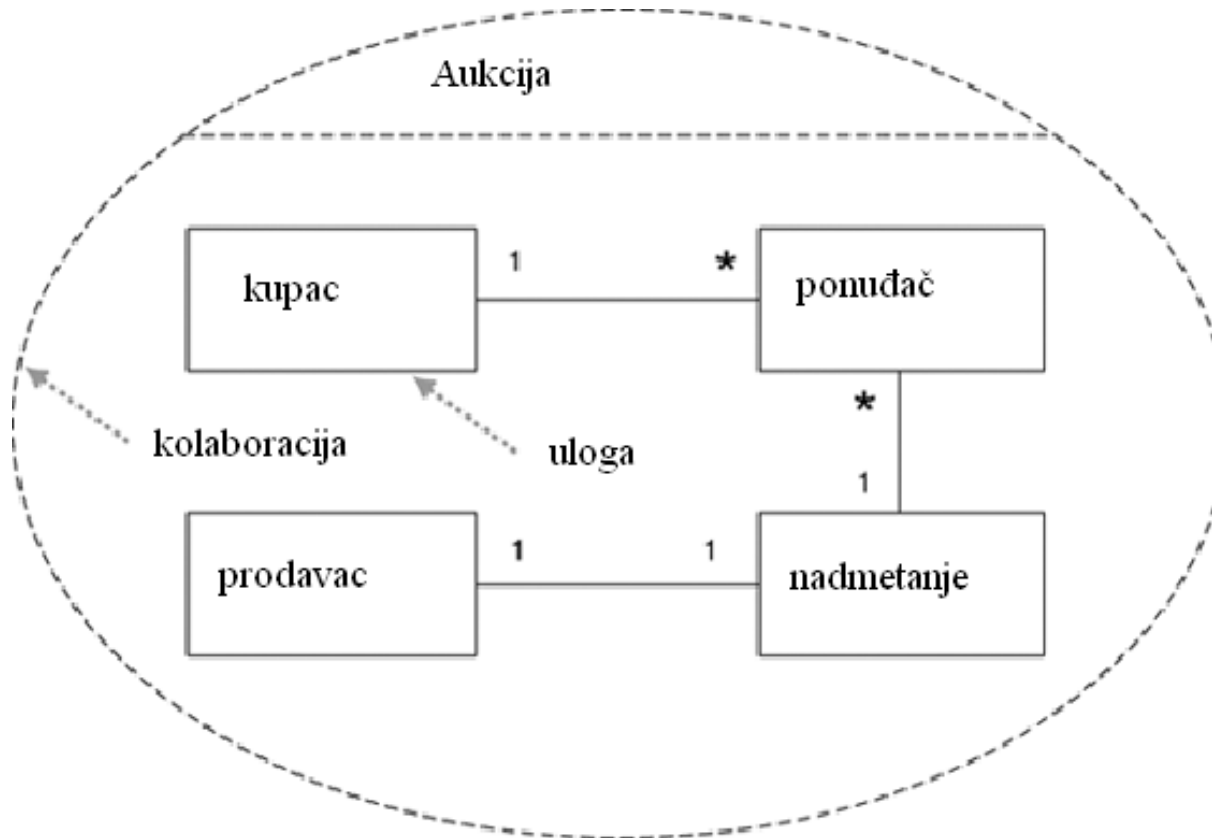
Ovi dijagrami se mogu povezati sa kompozitnim strukturama. Oni predstavljaju **dijagrame interakcija**, tj. Odnose se na dinamičke aspekte sistema koji se modeluje.

Kod dijagrama saradnje se ističe strukturna organizacija objekata koji šalju i primaju poruke.

Grafički, kolaboracioni dijagram je kolekcija čvorova i grana.

Na dijagramima se prikazuju objekti i njihove uloge, veze, poruke i redosledi.

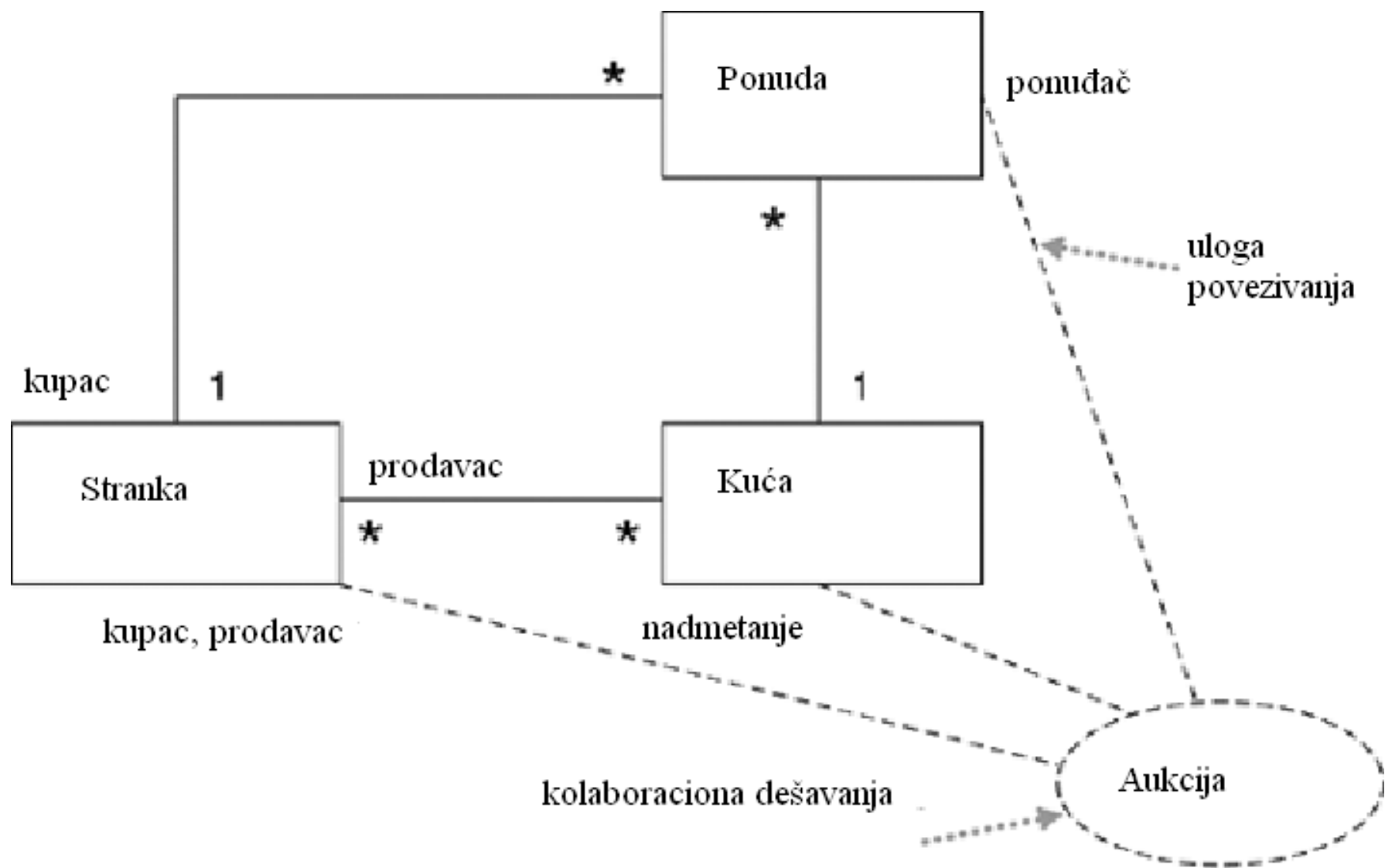
Na sledećoj slici je predstavljen klasni dijagram za aukciju.



U prethodnom primeru nije predstavljen pravi dijagram klasa (nisu ni prave oznake za klase). Ovde klase pre opisuju neke ULOGE koje će se realizovati ako se kolaboracija (saradnja) ostavri. Zato je sve zaokrugljeno elipsom nacrtanom isprekidanom linijom.

Ako želimo na pravi način da izrazimo saradnju (kolaboraciju), onda postavljamo kolaboraciona zbivanja na klasni dijagrama. Veza između klasa i kolaboracije pokazuje kako klase igraju različitu ulogu od kojih je svaka definisana u saradnji.

Jedan primer kolaboracionog dijagrama prikazan je na sledećoj slici.



Sekvencni dijagrami

Sekvencni dijagrami predstavljaju **interakcione dijagrame**. Oni pokazuju kako saraduje i kako se ponaša grupa učesnika u toku nekog scenarija. Sekvencni dijagrami obično prikazuju neke **učesnike (participants)** (to je novi pojam u UML 2, dok je u UML 1 korišćen pojam objekat, što učesnik najčešće i jeste) i poruke koje se razmenjuju među njima. Znači, oni služe za modelovanje dinamičkih aspekata nekog sistema.

Za razliku od kolaboracionih dijagrama (kod kojih se ističe strukturna organizacija učesnika), kod sekvencnih dijagrama se ističe **vremenski redosled poruka**.

Grafički, kod sekvencnih dijagrama učesnici se ređaju po x-osi, a poruke (poređane u vremenskom redosledu) po y-osi.

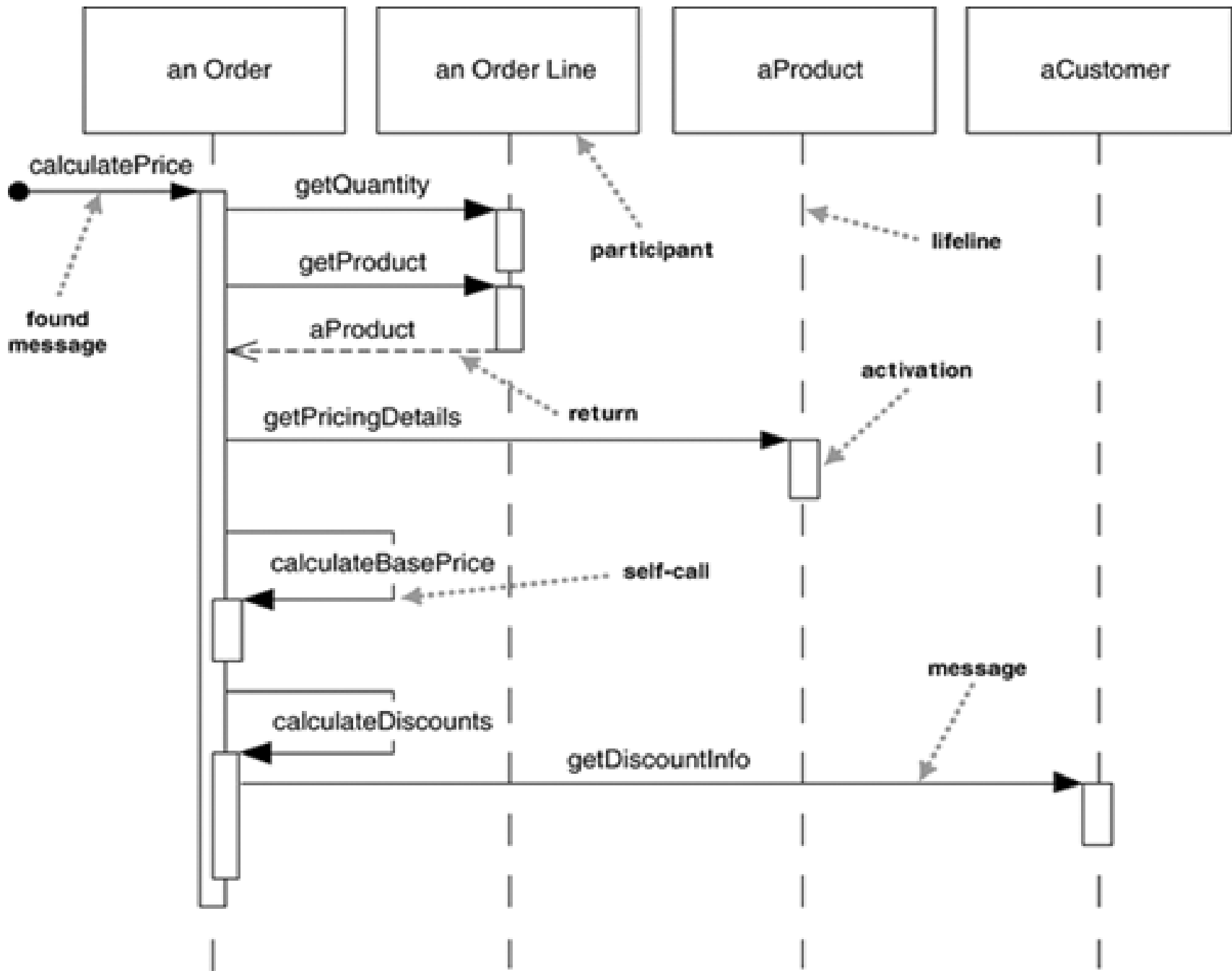
Učesnik, koji započinje interakciju, obično se stavlja na vrh s leve strane, a ostali se ređaju udesno. Zatim se unose poruke (na y-osi) koje ti učesnici šalju i primaju (od vrha ka dnu).

Sekvencni dijagrami se razlikuju od kolaboracionih i komunikacionih po dvema osobinama:

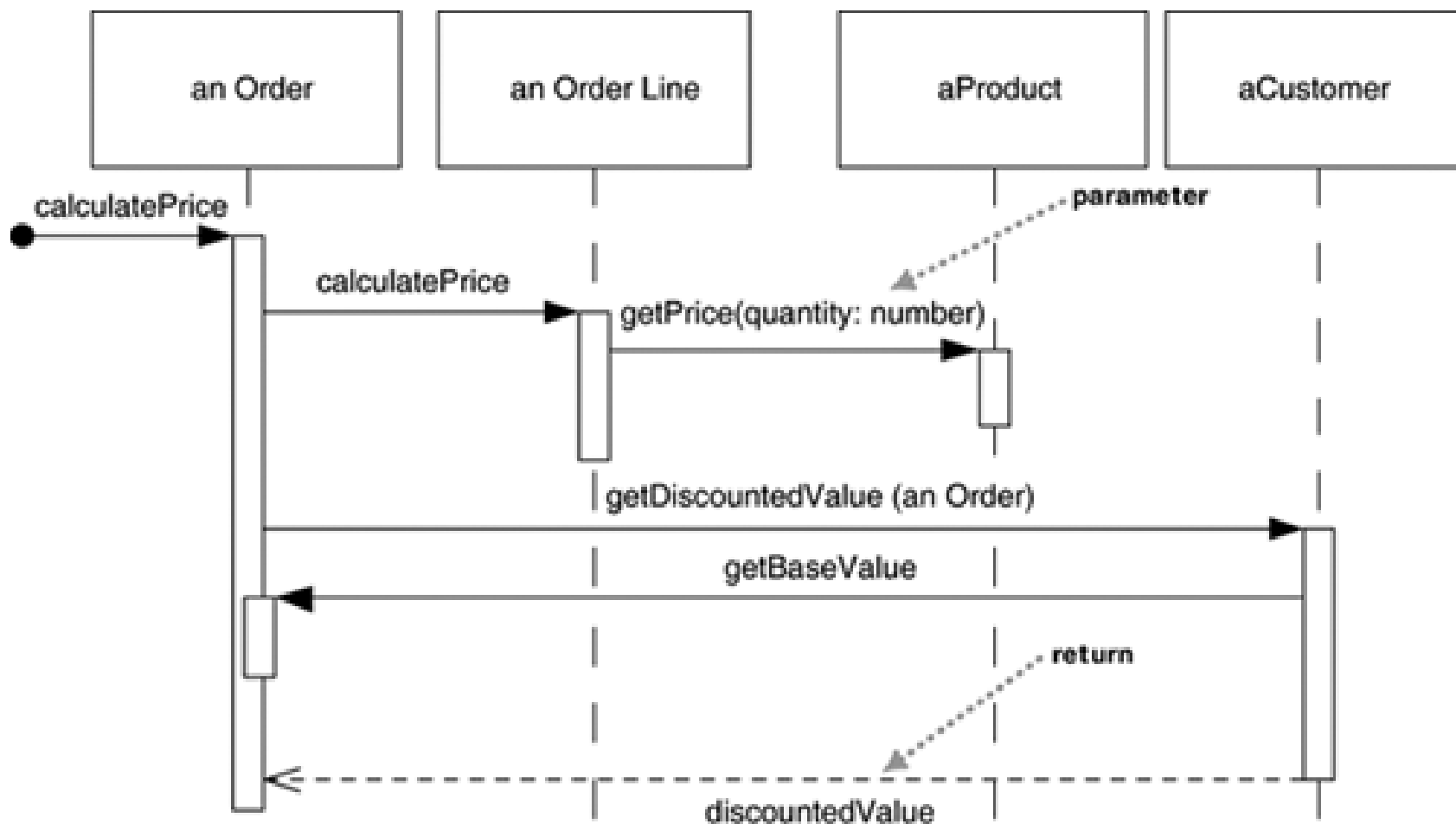
-Postoji linija života objekata – vertikalni crtkasta linija koja pokazuje da objekat postoji u nekom vremenskom periodu.

-Postoji fokus upravljanja – uzak dugačak pravougaonik koji pokazuje period vremena u kojem učesnici izvode neke akcije.

Na sledećoj slici je prikazan jedan od mogućih scenarija za obradu narudžbenice. Kod ovog scenarija zastupeljena je tzv. **centralizovana kontrola** (sve glavne poslove obavlja jedan učesnik, a ostali ga snabdevaju podacima).



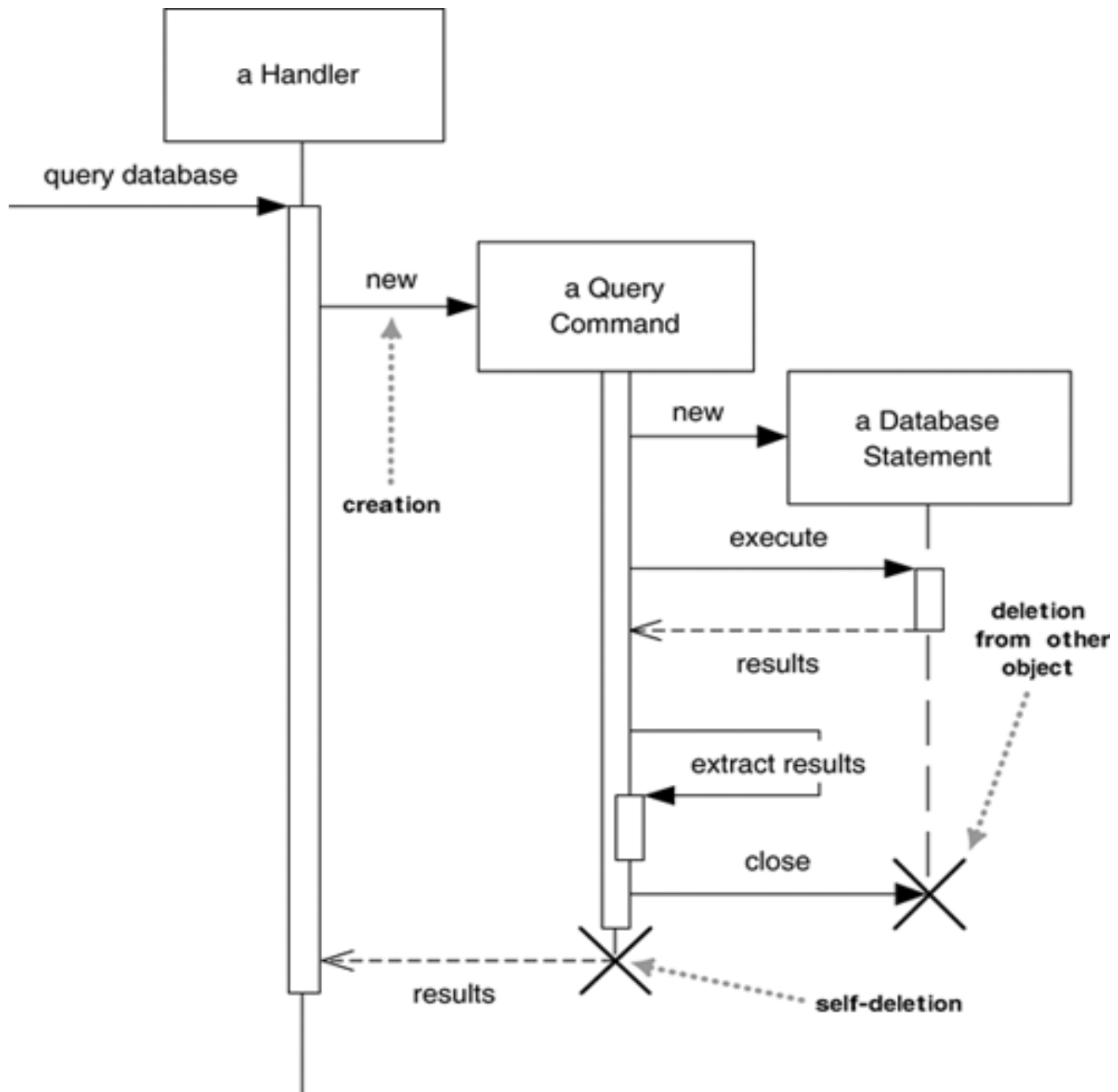
Na sledećoj slici je opisan nešto drugačiji scenario za isti zadatak. Ovde je zastupljena tzv. **distribuirana kontrola**. Ona je više u duhu OO-stila programiranja.



Preko sekvencnih dijagrama može da se prikaže **kreiranje i brisanje** učesnika. Kreiranje se naznačava strelicom (direktno u boks učesnika), a brisanje učesnika se označava velikim **X**. (Kod okolina sa skupljačem otpadaka brisanje nije toliko bitno, ali je korisno naznačiti ga u dijagramu.)

U ovakvom dijagramu učesnici ne postoje za sve vreme odvijanja scenarija pa nisu prikazani na istom nivou duž x-ose, već u vremenu nastajanja.

Ako učesnik, odmah nakon kreiranja, nešto radi, dijagram aktiviranja polazi neposredno iz boksa učesnika.



Sekvencni dijagrami nisu pogodni za opis ciklusa, grananja, ... - oni pre svega služe za vizuelizaciju interakcije među objektima! Za algoritamske kontrole pogodniji su dijagrami aktivnosti (ili sam kod programa !).

Međutim, algoritamski elementi mogu se prikazati i na sekvencnim dijagramima.

Za opis ciklusa i grananja koriste se tzv. **interakcioni okviri (novi pojam u UML 2.0!)** koji služe za specifično obeležavanje pojedinih delova sekvencnog dijagrama. Jedan okvir može sadržati više **fragmenta**. Fragment može imati **gard (zaštitni uslov)**. Fragment se izvršava jedino ako je gard tačan.

Sledi tablica sa objašnjenjem tekstualnih oznaka koje se koriste u interakcionim okvirima za opis raznih vrsta kontrola.

alt -- alternativni višestruki fragmenti – izvršavaju se samo oni čiji uslov je tačan.

opt – opcioni – izvršava se samo ako je navedeni uslov tačan; the fragment executes only if the supplied condition is true (ekvivalentan alt-operatoru sa jednim tokom).

par – paralelno – svi fragmenti se paralelno izvršavaju.

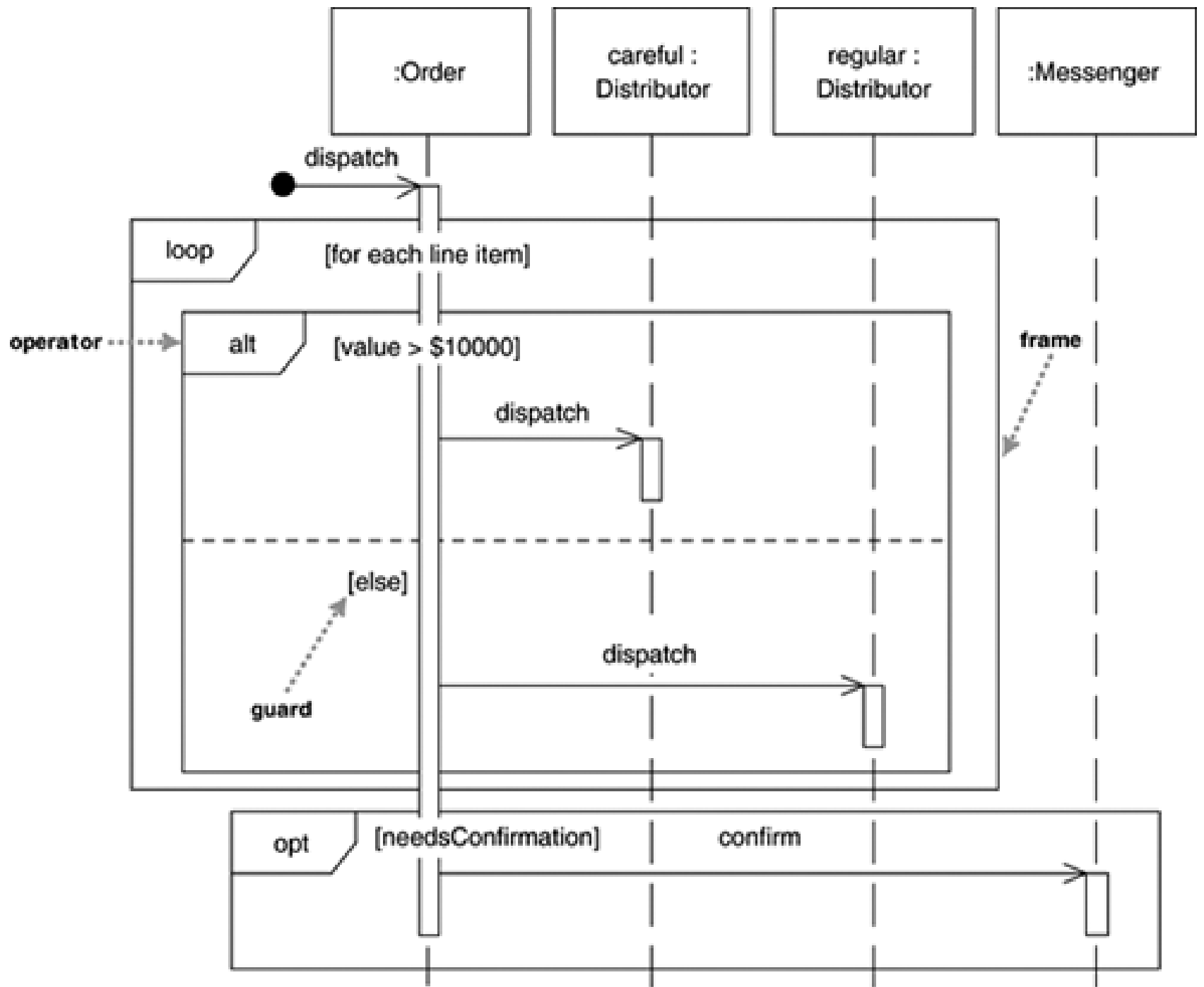
loop – ciklus – fragment se može izvršiti više puta i zaštitni uslov predstavlja osnovu iteracije.

region – kritični region – fragment može imati samo jedan tred koji se izvršava u celosti.

neg – negativno – fragment pokazuje nepravilnu interakciju.

ref – referenca – odnosi se na interakciju definisanu na još jednom dijagramu.

sd – sekvencni dijagram – koristi se (po potrebi) za zaokruživanje celog sekvencnog dijagrama.



Strelice u poslednjem i oprethodnim dijagramima su različito označene. Oznake strelica su bitne u verziji UML 2.

Popunjene strelice pokazuju **sinhronizovane pozive**, dok nepopunjene pokazuju **nesinhronizovane pozive** (u verziji UML 1.x korišćena je polustrelica!).

Ako je poslata sinhronizovana poruka, pošaljalac čeka dok se poruka ne realizuje. Ako je neko poslao nesinhronizovanu poruku, može nastaviti obradu i ne mora da čeka na odgovor.

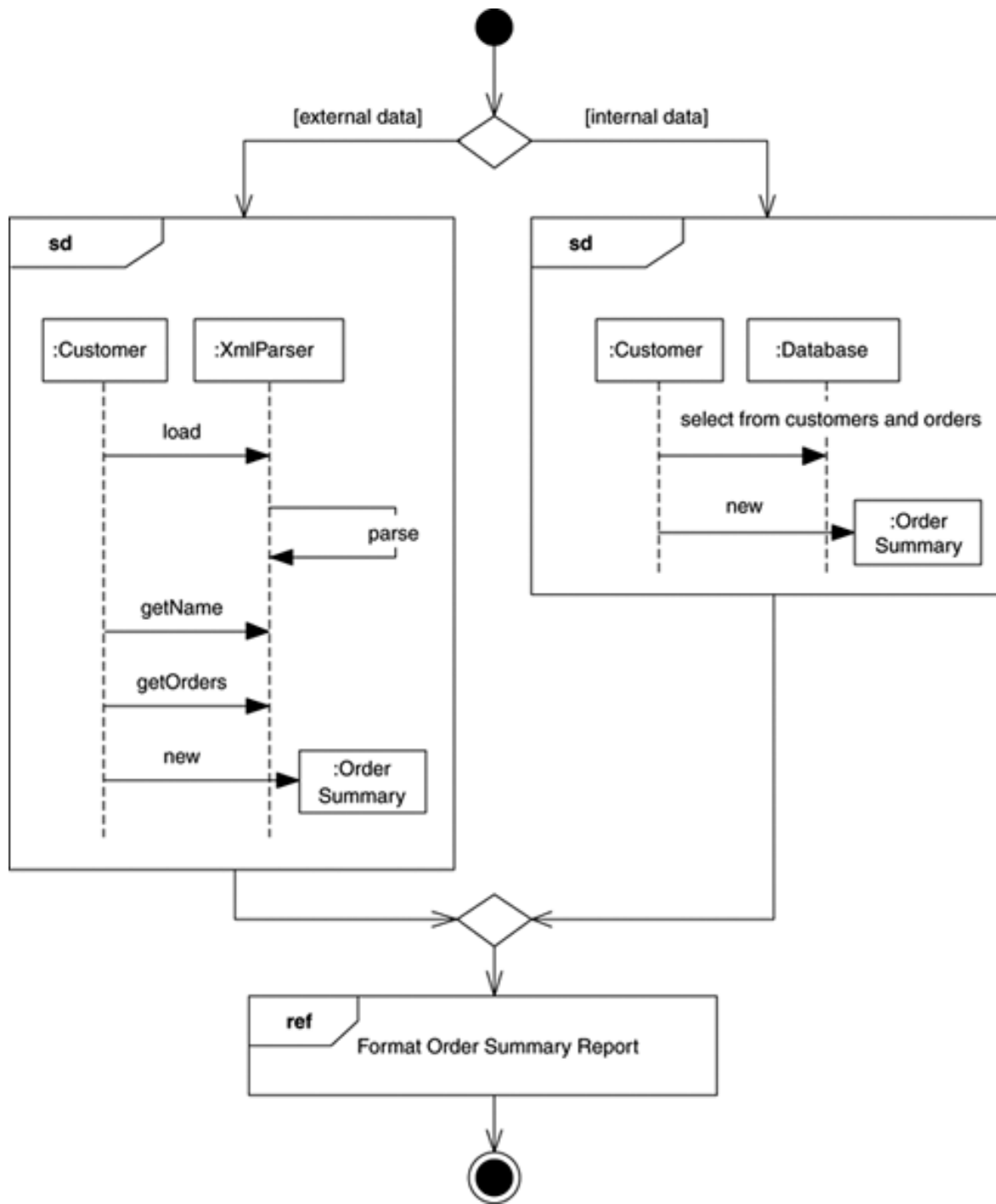
Nesinhronizovani pozivi karakterišu multitredovne aplikacije i na porukama zasnovanu razmenu. Aplikacije sa asinhronizovanim pozivima su teže za debugovanje.

Opštepregledni interakcioni dijagrami mogu se formirati kombinovanjem dijagrama aktivnosti i skvencnih dijagrama.

Mogu se zamisliti kao dijagrami aktivnosti u kojima su aktivnosti zamenjene malim skvencnim dijagramima ili kao skvencni dijagrami razbijeni na dijegrame aktivnosti da bi se prikazala kontrola toka.

Na sledećoj slici je primer jednog opštepreglednog interakcionog dijagrama.

I ova vrsta kombinacije je nova u verziji UML 2, tako da će se tek ubuduće videti da li će biti opšteprihvaćena.



Vremenski dijagrami

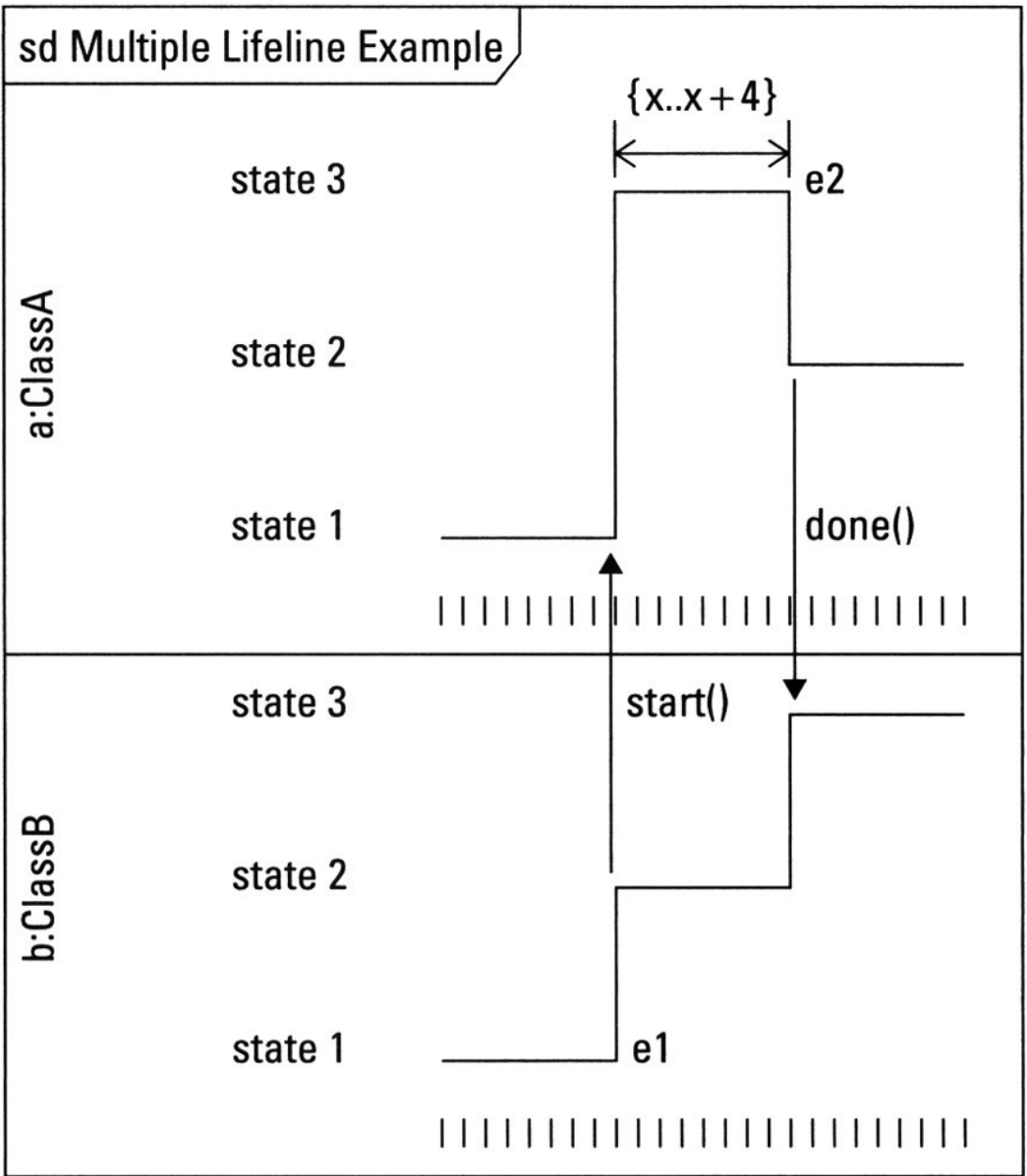
To je još jedna vrsta interakcionih dijagrama kod kojih je akcenat na vremenskim ograničenjima. Mogu se odnositi na jedan objekat, ali češće se odnose na više njih.

Promena stanja u vremenskim dijagramima može se opisivati linijama ili površinskim zonama.

Na sledećim slikama prikazana su 2 vremenska dijagrama u kojima se koriste ova 2 načina za prikaz vremenskih dešavanja.

U 1. primeru događaj e_1 izaziva promenu stanja u objektu b . Objekt b reaguje upućivanjem poruke `start()` objektu a . Kasnije, događaj e_2 izaziva promenu stanja objekta a , koji šalje poruku `done()` utičući na promenu stanja objekta b .

U 2. primeru površinska zona opisuje promene stanja u vremenu.



sd Single Lifeline Example

Event



event end date
has passed

